CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Quality Evaluation of Scenario-Tree Generation Methods for Solving Stochastic Programming Problem

Julien Keutchayan
Michel Gendreau
Antoine Saucier

March 2017

CIRRELT-2017-17

Bureaux de Montréal :
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Université Laval · McGill · Concordia · ÉTS · UQÀM · HEC Montréal · Polytechnique Montréal · Université de Montréal

Quality Evaluation of Scenario-Tree Generation Methods for Solving Stochastic Programming Problem†

Julien Keutchayan[1,2,*], Michel Gendreau[1,2], Antoine Saucier[2]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Mathematics and Industrial Engineering, Polytechnique Montréal, P.O. Box 6079, Station Centre-Ville, Montréal, Canada H3C 3A7

Abstract. This paper addresses the generation of scenario trees to solve stochastic programming problems that have a large number of possible values for the random parameters (possibly infinitely many). For the sake of the computational efficiency, the scenario trees must include only a finite (rather small) number of scenarios, therefore, it provides decisions only for some values of the random parameters. To overcome the resulting loss of information, we propose to introduce an *extension procedure*. It is a systematic approach to interpolate and extrapolate the scenario-tree decisions to obtain a decision policy that can be implemented for any value of the random parameters at little computational cost. To assess the quality of the scenario-tree generation method and the extension procedure (STGM-EP), we introduce three generic quality parameters that focus on the quality of the decisions. We use these quality parameters to develop a framework that will help the decision-maker to select the most suitable STGM-EP for a given stochastic programming problem. We perform numerical experiments on two case studies. The quality parameters are used to compare three scenario-tree generation methods and three extension procedures (hence nine couples STGM-EP). We show that it is possible to single out the best couple in both problems, which provides decisions close to optimality at little computational cost.

Keywords: Stochastic programming, scenario tree, decision policy, out-of-sample evaluation

Acknowledgements. This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grants RGPIN-2015-04696 and PIN-115965. This support is gratefully acknowledged.

†Revised version of the CIRRELT-2016-46.

Results and views expressed in this publication are the sole responsibility of the authors and do not necessarily reflect those of CIRRELT.

Les résultats et opinions contenus dans cette publication ne reflètent pas nécessairement la position du CIRRELT et n'engagent pas sa responsabilité.

---

* Corresponding author: Julien.Keutchayan@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2017

© Keutchayan, Gendreau, Saucier and CIRRELT, 2017

1 Introduction

Stochastic programming is a mathematical programming framework used to formulate and solve sequential decision-making problems under uncertainty. It relies on the assumption that the probability distribution of the random parameters is known (possibly inferred from data), and that the information about these parameters becomes available stage by stage. If the distribution is supported by a finite number of points, small enough for a tractable computation, then all the random outcomes can be represented in a so-called *scenario tree*, and solving the stochastic program on the scenario tree provides the optimal decisions for every outcome. In this context, stochastic programming has proved to be a powerful framework to solve problems in energy, transportation, logistic, finance, etc.; see, e.g., Wallace and Fleten (2003), Schultz et al. (2003), Yu et al. (2003), Louveaux (1998) and Powell and Topaloglu (2003). The situation becomes more complicated if the random parameters take a large (possibly infinite) number of values, since stochastic programming problems are then large-scale optimization problems (possibly infinite dimensional problems) that are typically impossible to solve analytically or computationally in a reasonable time. In that case, the scenario tree is built with a finite subset of scenarios, obtained by discretizing the stochastic process that models the random parameters across the stages. Many discretization schemes for generating scenario trees have been developed in the literature; we will cite some important references in Section 1.2. The scenario-tree generation method enables the decision-maker to obtain estimates of the optimal value and the optimal solutions of the stochastic program, but two questions remain open for the decision-maker:

- How to implement the optimal solutions that are scenario dependent? Apart from the first-stage decisions, which are not scenario dependent, all subsequent stage decisions depend on the scenarios, and therefore they may not be implementable if the real-world realization of the stochastic process does not coincide with a scenario in the tree.
- How to tell which method provides the best quality decisions for a given problem? Methods are typically built from mathematical results on the optimal-value error (consistency, rate of convergence, etc.), but it is unclear whether a good optimal-value estimate systematically implies good quality decisions. Additionally, the claimed effectiveness may be guaranteed under assumptions that are not fulfilled in practice. Also, it may hide some unknown quantities (e.g., the implied constant in a big-O notation for the rate of convergence) that prevents the decision-maker from knowing with certainty that the best method from a theoretical point of view will be the most efficient when put into practice.

In this paper, we show that both questions are inherently linked and we propose a mathematical framework that answers both of them.

The remainder of this paper is organized as follows: In Section 1.1 and 1.2, we introduce the notation to describe the stochastic programming problem and the scenario-tree formulation; this notation is summarized in Appendix A. In Section 1.3, we describe with more details the motivation of our approach. In Section 2, we develop the quality evaluation framework, and we provide in Section 3 the statistical tools (estimators, confidence intervals) to put it into practice. We present actual extension procedures in Section 4, and we apply them in the two case studies in Section 5. Finally, Section 6 concludes the paper.

1.1 Stochastic programming problem formulation

We consider a stochastic programming problem with a time horizon $T \in \mathbb{N}^*$ and integer time stages t ranging from 0 to T. The stagewise evolution of the random parameters is represented by a

stochastic process $\boldsymbol{\xi} = (\boldsymbol{\xi}_1, ..., \boldsymbol{\xi}_T)$, defined on a probability space $(\Omega, \mathcal{A}, \mathbb{P})$, where $\boldsymbol{\xi}_t$ is a random vector with d_t components that represent the random parameters revealed in period $(t-1, t)$. We define $\boldsymbol{\xi}_{..t} := (\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_t)$ the partial stochastic process up to stage t, and we denote the supports of $\boldsymbol{\xi}_t$, $\boldsymbol{\xi}_{..t}$, and $\boldsymbol{\xi}$ by Ξ_t, $\Xi_{..t}$, and Ξ, respectively. Throughout this paper, random quantities are always written in bold font, while their realizations are written with the same symbols in normal font.

At each stage $t \in \{1, \ldots, T\}$, the decisions must be based only on the information available at this stage in order to be *non-anticipative*. Thus, the stage-t decision function, denoted by x_t, is defined as

$$\begin{aligned} x_t : \Xi_{..t} &\to \mathbb{R}^{s_t} \\ \xi_{..t} &\mapsto x_t(\xi_{..t}), \end{aligned} \tag{1}$$

where s_t is the number of decisions to be made at stage t (for the sake of clarity, we consider that $s_t = s$ and $d_t = d$ for all t). The decisions at stage 0 are represented in a vector $x_0 \in \mathbb{R}^s$. We assume that each decision function belongs to an appropriate space of measurable functions, e.g., the space $\mathcal{L}_p(\Xi_{..t}; \mathbb{R}^s)$ of p-integrable functions for $p \in [1, +\infty]$. The *decision policy* (or simply *policy*) is denoted by x and is the collection of all the decision vector/functions from stage 0 to stage T, i.e., $x = (x_0, x_1, \ldots, x_T) \in \mathbb{R}^s \times \Pi_{t=1}^{T} \mathcal{L}_p(\Xi_{..t}; \mathbb{R}^s)$ or equivalently

$$x(\boldsymbol{\xi}) = (x_0, x_1(\boldsymbol{\xi}_1), \ldots, x_t(\boldsymbol{\xi}_{..t}), \ldots, x_T(\boldsymbol{\xi})). \tag{2}$$

The set of feasible decision vectors (or simply feasible set) is denoted by X_0 at stage 0 and by $X_t(x_{..t-1}(\xi_{..t-1}); \xi_{..t})$ at stage $t \in \{1, \ldots, T\}$; the latter notation emphasizes that it may depend on the realization $\xi_{..t} \in \Xi_{..t}$ and on the decisions $x_{..t-1}(\xi_{..t-1}) := (x_0, ..., x_{t-1}(\xi_{..t-1}))$ prior to t. A decision policy is *feasible* if it yields a feasible decision vector at every stage with probability one. We assume that the space of feasible policies is nonempty and that the decisions made at each stage do not alter the probability distribution of the stochastic process. We emphasize that our modelization can include integrity constraints.

We introduce a revenue function $q(x(\xi); \xi)$ that represents the total revenues obtained from stage 0 to T for a policy x and a realization ξ of the stochastic process. We are interested in the dependence of the first moment of $q(x(\boldsymbol{\xi}); \boldsymbol{\xi})$ with respect to the decision policy, i.e., in the functional $Q(x) := \mathbb{E}[q(x(\boldsymbol{\xi}); \boldsymbol{\xi})]$; we suppose that $Q(x)$ is well-defined for any feasible policy x.

The stochastic programming problem consists in finding a feasible and non-anticipative policy that maximizes $Q(\cdot)$, which means finding x^* of the form (2) satisfying

$$Q(x^*) = \max_{x=(x_0,\ldots,x_T)} \mathbb{E}[q(x(\boldsymbol{\xi}); \boldsymbol{\xi})] \tag{3}$$

$$s.t. \quad x_0 \in X_0; \tag{4}$$

$$x_1(\boldsymbol{\xi}_1) \in X_1(x_0; \boldsymbol{\xi}_1), \ w.p.1; \tag{5}$$

$$x_t(\boldsymbol{\xi}_{..t}) \in X_t(x_{..t-1}(\boldsymbol{\xi}_{..t-1}); \boldsymbol{\xi}_{..t}), \ w.p.1, \ \forall t \in \{2, \ldots, T\}. \tag{6}$$

Constraints (5) and (6) hold with probability one (w.p.1). The policy x^* is called an *optimal decision policy* and $Q(x^*)$ is the *optimal value* of the stochastic programming problem. Some additional conditions should be added to ensure that there exists at least one optimal decision policy; see, e.g., Rockafellar and Wets (1974).

1.2 Scenario tree and scenario-tree deterministic program

In most problems the optimal value $Q(x^*)$ and the optimal decision policy x^* are difficult to compute exactly, or approximately within a sufficiently small error, as shown in Dyer and Stougie (2006) and

Hanasusanto et al. (2016). For this reason, approximate solution methods have been developed, such as the large family of scenario-tree generation methods. We refer the reader to the following references for a general presentation on stochastic programming and solution methods: Birge and Louveaux (1997), Ruszczyński and Shapiro (2003), Schultz (2003), and Defourny et al. (2011).

A scenario-tree generation method builds a scenario-tree deterministic program from a finite subset of realizations of $\boldsymbol{\xi}$ (called *scenarios*). The scenarios are obtained through a discretization scheme, which can be performed using many possible techniques, and are organized in a tree structure to approximate the stagewise evolution of information (called *filtration*) of the stochastic process. Some of the most popular works on scenario generation are: Shapiro and Homem-de Mello (1998) and Mak et al. (1999) on the Monte Carlo method; Pennanen and Koivu (2002), Drew and Homem-de Mello (2006), and Leövey and Römisch (2015) on integration quadrature and quasi-Monte Carlo; Høyland and Wallace (2001) and Høyland et al. (2003) on moment-matching; Pflug (2001), Pflug and Pichler (2012), and Pflug and Pichler (2015) on optimal quantization; Dupačová et al. (2003) and Heitsch and Römisch (2009) on scenario reduction; Frauendorfer (1996) and Edirisinghe (1999) on bound-based approximations; Chen and Mehrotra (2008) and Chen et al. (2015) on sparse grid quadrature rules.

All the above methods provide a procedure to generate scenarios from $\boldsymbol{\xi}$, but only a few also provide a systematic approach to generate a *tree structure*, i.e., to organize the scenarios in a structure with branchings at every stage. In most cases, the choice of a tree structure is left to the decision-makers themselves, who may choose it empirically. Throughout this paper, we use the term *scenario-tree generation method* to name a procedure that generates a set of scenarios organized in a tree structure; this includes the case where the structure is chosen beforehand by the decision-maker. The remainder of this section introduces the notation for the scenario tree and the scenario-tree deterministic program.

A scenario tree is a rooted tree structure $\mathcal{T} = (\mathcal{N}, \mathcal{E})$, with (finite) node set $\mathcal{N}$, edge set $\mathcal{E}$, and root node n_0. The structure is such that T edges separate the root from any of the leaves. We denote by $C(n)$, $a(n)$, and $t(n)$, respectively, the children nodes of n, the ancestor node of n, and the stage of n (i.e., the number of edges that separate n from n_0). We also denote $\mathcal{N}^+ := \mathcal{N} \setminus \{n_0\}$ and $\mathcal{N}_t := \{n \in \mathcal{N} \,|\, t(n) = t\}$. Each node $n \in \mathcal{N}^+$ carries a discretization point ζ^n of $\boldsymbol{\xi}_{t(n)}$ and a weight $w^n > 0$. The latter represents the weight of n with respect to its sibling nodes. The weight of n with respect to whole scenario tree, denoted by W^n, is the product of all w^m for m on the path from n_0 to n. We emphasize that we let the weights be any positive real values to cover a large setting of discretization schemes. We denote by $\zeta^{\cdot n}$ the sequence of discretization points on the path from n_0 to n; hence $\zeta^{\cdot n}$ is a discretization point of $\boldsymbol{\xi}_{\cdot t(n)}$.

The scenario-tree approach proceeds as follows: the vector $\widehat{x}^n \in \mathbb{R}^s$ is the decision at node $n \in \mathcal{N}$ and the sequence $\widehat{x}^{\cdot n} := (\widehat{x}^{n_0}, \ldots, \widehat{x}^n)$ denotes the decision vectors on the path from n_0 to n. The scenario-tree deterministic program is written as

$$\widehat{Q}^* := \max_{\{\widehat{x}^n : n \in \mathcal{N}\}} \sum_{l \in \mathcal{N}_T} W^l \, q(\widehat{x}^{\cdot l}; \zeta^{\cdot l}) \tag{7}$$

$$s.t. \quad \widehat{x}^{n_0} \in X_0; \tag{8}$$

$$\widehat{x}^n \in X_1\big(\widehat{x}^{n_0}; \zeta^n\big), \ \forall n \in \mathcal{N}_1; \tag{9}$$

$$\widehat{x}^n \in X_t\big(\widehat{x}^{\cdot a(n)}; \zeta^{\cdot n}\big), \ \forall n \in \mathcal{N}_t, \ \forall t \in \{2, \ldots, T\}, \tag{10}$$

The optimal decision vector at each node is denoted by $\widehat{x}^{n*}$ and, for convenience, we define $\widehat{x}^* := \{\widehat{x}^{n*} \,|\, n \in \mathcal{N}\}$ and we refer to it as the *tree optimal policy*.

We emphasize that if the stochastic program cannot be solved exactly, which is our case of interest in this paper, then the outputs $\widehat{x}^*$ and $\widehat{Q}^*$ of the scenario-tree approach are approximations

of x^* and $Q(x^*)$, respectively.

1.3 Motivations and extension procedure formulation

Scenario trees have proved to be a useful approach for a wide class of stochastic programming problems (see the references in the Introduction). However, as pointed out in Ben-Tal et al. (2009), this approach fails to provide decisions for all values of the random parameters. The reason is that the decisions at stage t are only available for the set $\{\zeta^{\cdot n} \,|\, n \in \mathcal{N}_t\}$, which is a proper subset of $\Xi_{\cdot t}$. If the stochastic process has a continuous distribution, then the former set has probability zero, and therefore the real-world realization of the stochastic process never coincides with a scenario in the tree. In that case, only the stage-0 decision, which is not scenario dependent, can be implemented by the decision-maker. This raises the first question written in the Introduction: How to implement the optimal solutions that are scenario dependent?

An attempt to answer this question, proposed for instance in Kouwenberg (2001), Chiralaksanakul and Morton (2004), and Hilli and Pennanen (2008), consists in solving dynamically the scenario-tree deterministic program on a shrinking horizon in order to implement the stage-0 decision recursively at every stage. However, a drawback of this approach is its computational cost. It requires as many solutions as the total number of stages, and the procedure must be carried out all over again for each new realization ξ. With this approach, it can be computationally costly to perform an out-of-sample test, which is an evaluation of the tree decisions on a set of scenarios directly sampled from $\boldsymbol{\xi}$, since it is typically required to test the decisions on thousands of realizations for a reliable accuracy. Therefore, a satisfactory answer to the first question would be to find a way to provide decisions for *any* value of the random parameters, in a manner that allows a thorough out-of-sample test.

The second question raised in the introduction is concerned with the choice of the scenario-tree generation method. Methods are usually developed with the goal to control the optimal-value error $|Q(x^*) - \widehat{Q}^*|$. But as far as the decisions are concerned, it is unclear whether a small value of the error always implies a tree optimal policy $\widehat{x}$ close to the optimal decision policy x^*. Additionally, the notion of closeness between the two policies is in itself difficult to define, because $\widehat{x}$ is a finite set of vectors whereas x^* is a sequence of functions. For this reason, the focus is sometimes made on controlling the distance between the two stage-0 decision vectors $\widehat{x}^{n_0}$ and x_0^*. This is done for instance in Pennanen (2005), where the scenario trees are generated in a way that guarantees the convergence of $\widehat{x}^{n_0}$ toward x_0^* as the number of scenarios increases. However, such approaches address only the quality of the stage-0 decisions. They ignore the decisions in the following stages, as if those decisions were irrelevant or irremediably out of reach for an evaluation. We do not believe so.

In this paper, we completely depart from the view of the references above. From the tree optimal policy $\widehat{x}$, we intend to recover a decision policy of the form (2) in order to treat the decisions of the scenario tree as a candidate solution of the stochastic programming problem. We do so as follows: after solving the program (7)-(10), we extrapolate and interpolate the tree optimal decisions outside the set of the scenarios. We refer to this as an *extension procedure* and to the resulting policy as an *extended tree policy*. The extended tree policy (formalized in Definition 1.1) is defined over all possible realizations of the stochastic process, and it coincides with the tree optimal policy on the scenarios of the tree.

Definition 1.1. *Let $\widehat{x}^* = \{\widehat{x}^{n*} \,|\, n \in \mathcal{N}\}$ be a tree optimal policy. An* extended tree policy *for $\widehat{x}^*$ is a decision policy $\widetilde{x} = (\widetilde{x}_0, \ldots, \widetilde{x}_T)$, where $\widetilde{x}_0 = \widehat{x}^{n_0*}$ and for every $t \in \{1, \ldots, T\}$, $\widetilde{x}_t$ is defined as in* (1) *and satisfies*

$$\widetilde{x}_t(\zeta^{\cdot n}) = \widehat{x}^{n*}, \quad \textit{for all } n \in \mathcal{N}_t. \tag{11}$$

The extension procedure enables a thorough out-of-sample test of the policy, because the decisions are available merely through the evaluation of a function at a particular point, which can be carried out many times at little cost. Since the extended tree policy $\widetilde{x}$ is a candidate solution of the stochastic program, it can be compared with the optimal policy x^*. A natural comparison criterion, which is highly relevant for the decision-maker, is to compare the value $Q(\widetilde{x})$ with $Q(x^*)$. Although this idea provides the basis for our quality evaluation framework, we show that it must be addressed with care because the extended tree policy may not satisfy the feasibility requirement.

Our quality evaluation framework is based on three quality parameters that will enable the decision-maker to compare couples of scenario-tree generation method and extension procedure (STGM-EP) in order to select the best one for a given problem. We refer to Kaut and Wallace (2007) for a discussion on the evaluation of scenario-tree generation methods for two-stage problems, which provided the inspiration for this paper. We also refer to the works of Defourny et al. (2013) where some extension techniques are introduced using regression tools from machine learning. However, their approach and ours differ in several aspects, the main one being their desire to select the best *policy*, while we want to select the best *method* (i.e., the whole family of policies that can be obtained by the method; see Remark 1.1). The reason why we focus on methods rather than on policies lies in the way scenario trees are used in practice. Very often, problems are solved regularly with different data to infer the random parameters, and therefore a new policy must be computed for each new data set. In that case, our quality evaluation framework requires to run the selection test just on one data set. Then, the selected STGM-EP can be used to compute a policy for each new data set without additional comparison tests. We will discuss this point in more details after the definition of the selection criterion in Section 2.3.

Remark 1.1. Although the framework developed in this paper can be applied to any scenario-tree generation method, the future mathematical developments require to differentiate two categories of methods. On the one hand, the *stochastic* methods use some random sampling techniques to build scenario trees. The stochasticity implies that every output scenario tree is different when the method is carried out multiple times. On the other hand, the *deterministic* methods generate a unique scenario tree (i.e., a unique tree structure, set of discretization points and weights).

Stochastic methods are more difficult to assess. Since different scenario trees imply different tree optimal policies, and therefore different extended tree policies, we need to be able to consider somehow all the possible extended tree policies that a stochastic method may yield. A way to do so is to see all these policies as the realizations of a *random* extended policy $\widetilde{\boldsymbol{x}}$, the latter is denoted in bold font to distinguish it from a particular realization $\widetilde{x}_\omega$ (see Figure 1). Therefore, by $\widetilde{\boldsymbol{x}}$ we denote the family of all decision policies that are obtained in a random manner by a particular STGM-EP.

Since the goal of this paper is to assess the quality of methods, we shall focus on studying the quality of $\widetilde{\boldsymbol{x}}$ rather than a specific realization of it. To this end, we will denote by $\mathbb{E}_{\widetilde{\boldsymbol{x}}}[\cdot]$ the expectation operator taken with respect to the probability measure of $\widetilde{\boldsymbol{x}}$. This measure is defined on the infinite dimensional space $\mathbb{R}^s \times \Pi_{t=1}^{T} \mathcal{L}_p(\Xi_{..t}; \mathbb{R}^s)$ of decision policies, which makes it a highly complicated mathematical object. However, we do not further develop this point in this paper because, as far as practitioners are concerned, the only important matter is that the probability measure of $\widetilde{\boldsymbol{x}}$ can be sampled by generating several scenario trees, and hence the expectation can be estimated by a finite sum. The statistical properties of such estimation will be studied in Section 3.

2 Quality parameters

In this section, we introduce the quality parameters that assess any scenario-tree generation method and extension procedure. We assume the following condition holds throughout this section:

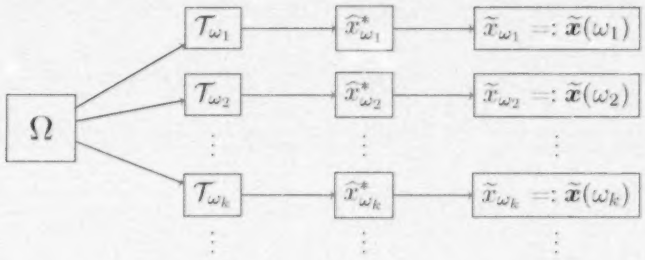


Figure 1: A *stochastic* STCM-EP yields a *random* extended tree policy $\widetilde{\boldsymbol{x}}$. As a random element, this policy can be seen as a map $\widetilde{\boldsymbol{x}} : \Omega \to \mathbb{R}^s \times \Pi_{t=1}^T \mathcal{L}_p(\Xi_{..t}; \mathbb{R}^s)$ obtained through the composition of several transformations represented by the arrows.

C1. The stochastic program (3)-(6) and the scenario-tree deterministic program (7)-(10) each have an optimal decision policy.

The framework developed in the section works for both stochastic and deterministic methods. However, for the sake of conciseness, it is expressed for stochastic methods only. The equivalent results for the deterministic ones are easy to deduce by removing the expectation $\mathbb{E}_{\widetilde{\boldsymbol{x}}}[\cdot|$ and by substituting $\widetilde{\boldsymbol{x}}$ with its unique realization $\widetilde{x}$.

2.1 Probability of feasibility and conditional revenues

It follows from Definition 1.1 that an extended tree policy yields feasible decisions at the root node and for any realization $\xi_{..t}$ that coincides with a discretization sequence $\zeta^{\cdot\cdot n}$ for a node $n \in \mathcal{N}_t$ in the scenario tree. For any other realization the feasibility is not guaranteed and will depend on the considered STCM-EP. The first two features of the extended tree policy that we want to assess is its probability of feasibility at every stage and its conditional revenues given the feasibility.

Consider a random extended tree policy $\widetilde{\boldsymbol{x}}$, and let $\widetilde{x}$ be a realization of $\widetilde{\boldsymbol{x}}$. The subset of $\Xi_{..t}$ on which $\widetilde{x}$ provides feasible decisions from stage 0 to t, denoted by $\widetilde{\Xi}_{..t}(\widetilde{x})$, is defined as

$$\widetilde{\Xi}_1(\widetilde{x}) = \big\{\xi_1 \in \Xi_1 \,|\, \widetilde{x}_1(\xi_1) \in X_1(\widetilde{x}_0; \xi_1)\big\}, \tag{12}$$

and

$$\widetilde{\Xi}_{..t}(\widetilde{x}) = \big\{\xi_{..t} \in \Xi_{..t} \,\big|\, \xi_{..t-1} \in \widetilde{\Xi}_{..t-1}(\widetilde{x}),\ \widetilde{x}_t(\xi_{..t}) \in X_t\big(\widetilde{x}_{..t-1}(\xi_{..t-1}); \xi_{..t}\big)\big\}, \tag{13}$$

for each $t \in \{2, \ldots, T\}$. Thus, the probability that $\widetilde{x}$ provides feasible decisions from stage 0 to t is $\mathbb{P}[\boldsymbol{\xi}_{..t} \in \widetilde{\Xi}_{..t}(\widetilde{x})]$. When considering the random policy $\widetilde{\boldsymbol{x}}$, the set $\widetilde{\Xi}_{..t}(\widetilde{\boldsymbol{x}}(\omega))$ varies depending on the outcome $\omega \in \Omega$ (see Figure 1). Taking into account the randomness of $\widetilde{\boldsymbol{x}}$ leads to the following definition of the quality parameters $p(t)$ and CR.

Definition 2.1. *(i) The probability $p(t)$ that a random extended tree policy $\widetilde{\boldsymbol{x}}$ yields feasible decisions up to stage $t \in \{0, \ldots, T\}$ is given by $p(0) = 1$ and*

$$p(t) = \mathbb{P}_{(\boldsymbol{\xi}, \widetilde{\boldsymbol{x}})}[\boldsymbol{\xi}_{..t} \in \widetilde{\Xi}_{..t}(\widetilde{\boldsymbol{x}})]. \tag{14}$$

(ii) The conditional revenues CR *obtained with $\widetilde{\boldsymbol{x}}$ when it yields feasible decisions up to the end of the optimization horizon is given by*

$$\mathrm{CR} = \mathbb{E}_{(\boldsymbol{\xi}, \widetilde{\boldsymbol{x}})}[q(\widetilde{\boldsymbol{x}}(\boldsymbol{\xi}); \boldsymbol{\xi}) \,|\, \boldsymbol{\xi} \in \widetilde{\Xi}_{..T}(\widetilde{\boldsymbol{x}})]. \tag{15}$$

The value CR *is well-defined provided that $p(T) > 0$.*

The sequence $(p(0), p(1), \ldots, p(T))$, non-increasing by definition of $\widetilde{\Xi}_{..t}(\widetilde{\boldsymbol{x}})$, provides information about the stagewise evolution of the size of the stage-t feasible region $\widetilde{\Xi}_{..t}(\widetilde{\boldsymbol{x}})$ (as a sequence of numbers ranging from 0 to 1) embedded in the support $\Xi_{..t}$.

Although CR is a natural quantity to compute, its interpretation can be tricky. By definition of the conditional expectation as a ratio of an expectation and a probability, the values of CR are inherently linked with those of $p(T)$. If $p(T)$ is less than one, then the conditional revenues are computed on a subset of random parameters, and therefore their values can be larger than the optimal ones $Q(x^*)$. Typically, it will be observed in the numerical experiments that the lower $p(T)$ the larger CR, which means that CR becomes almost irrelevant when $p(T)$ is much smaller than one, as it gives the expected revenues in a world where pessimistic scenarios are ignored. Conversely, if $p(T)$ is close to one, then CR is forced not to exceed $Q(x^*)$ too much (and in the limit $p(T) \to 1$, $Q(x^*)$ is an upper bound on CR), therefore its value is meaningful for the decision-maker.

When considering two STGM-EPs, denoted by A and B, a decision-maker will select A if the respective quality parameters satisfy:

$$p_A(T) > p_B(T) \quad \text{and} \quad \mathrm{CR}_A > \mathrm{CR}_B, \quad \text{with} \quad p_A(T) \geq \alpha, \tag{16}$$

where $\alpha \in (0, 1]$ is the feasibility threshold that the decision-maker considers as satisfactory. As we will see in the numerical experiments, the selection criterion (16) allows to put aside scenario-tree generation methods and extension procedures of poor quality. However, for the reason explained above, it is not always conclusive and it may not allow to single out the best method out of a group of good methods. In Section 2.3, we introduce a third quality parameter leading to a more robust selection criterion. To this end, we first address the feasibility restoration of the extended tree policy.

2.2 Feasibility restoration

In a real-world application, the extended tree policy $\widetilde{x}$ can be used all the way to the end of the optimization horizon provided the real-word realization ξ satisfies $\xi \in \widetilde{\Xi}_{..T}(\widetilde{x})$. If this condition does not hold, then there exists a stage $t^* \geq 0$ such that $\xi_{..t} \notin \widetilde{\Xi}_{..t}(\widetilde{x})$ for every $t > t^*$, and therefore the decision-maker has to find alternative feasible decisions from stage $t^* + 1$ to T. This is known in the literature as the *feasibility restoration* problem. A necessary condition for restoring the feasibility is that the decisions $(\widetilde{x}_0, \ldots, \widetilde{x}_{t^*}(\xi_{..t^*}))$ do not lead to an empty feasible sets from stage $t^* + 1$ to T. This is guaranteed if we assume that the following condition holds:

C2. The stochastic programming problem has a relatively complete recourse at every stage.

There are several approaches to address the feasibility restoration. In the works of Küchler and Vigerske (2010) and Defourny et al. (2013), the feasibility is restored by projecting the infeasible decision on the feasible set, which is done by solving non-linear optimization problems. In this paper, in order to proceed with the idea that the decisions must be available at little computational cost, we investigate the possibility that the decision-makers have the ability to fix any infeasibility by their own empirical knowledge on the problem. In other words, we assume that the following condition holds:

C3. The decision-maker possesses a *recourse policy*, obtained empirically, that always provides feasible decisions.

We model the recourse policy as a sequence $r = (r_1, \ldots, r_T)$, where the stage-t recourse function r_t takes a realization $\xi_{..t}$ and a sequence of decisions $x_{..t-1}(\xi_{..t-1})$ and yields a feasible decision vector,

i.e., $r_t(x_{..t-1}(\xi_{..t-1});\xi_{..t}) \in X_t(x_{..t-1}(\xi_{..t-1});\xi_{..t})$. We emphasize that the definition of r_t differs from the definition of x_t in (1), since r_t depends also on the previous decisions.

The implementation of $\widetilde{x}$ (if feasible) and r (otherwise) yields a new decision policy, called the *feasible extended tree policy*, that provides feasible decisions at every stage and for every realization of the stochastic process. Definition 2.2 provides its explicit construction.

Definition 2.2. *Let $\widetilde{x}$ be an extended tree policy and r a recourse policy. The* feasible extended tree policy $\overline{x}$ *resulting from $\widetilde{x}$ and r is given by $\overline{x}_0 = \widetilde{x}_0$ and recursively from $t = 1$ to $t = T$ by*

$$\overline{x}_t(\xi_{..t}) = \begin{cases} \widetilde{x}_t(\xi_{..t}) & \text{if } \xi_{..t} \in \widetilde{\Xi}_{..t}(\widetilde{x}), \\ r_t(\overline{x}_{..t-1}(\xi_{..t-1});\xi_{..t}) & \text{otherwise,} \end{cases} \tag{17}$$

where for $t = 1$ the term $\overline{x}_{..0}(\xi_{..0})$ corresponds to $\overline{x}_0$.

A stochastic STGM-EP yields a feasible extended tree policy that is random and is denoted by $\overline{\boldsymbol{x}}$ (see Remark 1.1 and Figure 1).

2.3 Distance between methods and selection criterion

The expected revenues obtained by implementing the feasible extended tree policy $\overline{x}$ is $Q(\overline{x}) = \mathbb{E}_{\boldsymbol{\xi}}[q(\overline{x}(\boldsymbol{\xi});\boldsymbol{\xi})]$. Since all realizations $\overline{x}$ of the random policy $\overline{\boldsymbol{x}}$ are feasible and non-anticipative, we have that

$$Q(\overline{\boldsymbol{x}}) = \mathbb{E}_{\boldsymbol{\xi}}[q(\overline{\boldsymbol{x}}(\boldsymbol{\xi});\boldsymbol{\xi})] \leq Q(x^*), \quad w.p.1. \tag{18}$$

We emphasize that the left-hand side of the inequality is a random variable, which is why the inequality holds *with probability one*. We see from (18) that every realization $\overline{x}$ of $\overline{\boldsymbol{x}}$ provides a lower bound $Q(\overline{x})$ of $Q(x^*)$. The nonnegative value $Q(x^*) - Q(\overline{x})$ provides a relevant measure of quality of $\overline{x}$. However, in general $Q(x^*)$ is not known, hence the computation may be done with $\widehat{Q}^*$ or an upper bound on $Q(x^*)$ rather than with $Q(x^*)$. This approach was used in the particular case of the Monte Carlo method for two-stage problems by Mak et al. (1999), and was developed further by Bayraksan and Morton (2009).

In this paper, we are interested in assessing the quality of methods, therefore, we must address the quality of $\overline{\boldsymbol{x}}$ rather than a specific realization $\overline{x}$. The inequality (18) still holds if we take the expectation of the left-hand side. This leads to the following definition of the distance $d(\overline{\boldsymbol{x}}, x^*)$ between the feasible extended tree policy $\overline{\boldsymbol{x}}$ and the optimal policy x^*.

Definition 2.3. *The distance $d(\overline{\boldsymbol{x}}, x^*)$ between the feasible extended tree policy $\overline{\boldsymbol{x}}$ and the optimal policy x^* of the stochastic program* (3)-(6) *is given by*

$$d(\overline{\boldsymbol{x}}, x^*) = Q(x^*) - \mathbb{E}_{\boldsymbol{x}}\big[Q(\overline{\boldsymbol{x}})\big] \geq 0. \tag{19}$$

This distance defines a concept of "optimal choice" for the selection of the scenario-tree generation method and the extension procedure, in the sense of the following proposition.

Proposition 2.4. *A scenario-tree generation method and an extension procedure yield a random policy $\overline{\boldsymbol{x}}$ that is optimal with probability one for the stochastic program* (3)-(6) *if and only if $d(\overline{\boldsymbol{x}}, x^*) = 0$.*

Proof. The proof is straightforward: we have that $d(\overline{\boldsymbol{x}}, x^*) = 0$ if and only if $\mathbb{E}_{\boldsymbol{x}}[Q(x^*) - Q(\overline{\boldsymbol{x}})] = 0$, and by the inequality (18), this is equivalent to $Q(x^*) = Q(\overline{\boldsymbol{x}})$ with probability one. By construction, the random policy $\overline{\boldsymbol{x}}$ is feasible and non-anticipative with probability one, which completes the proof. □

Thus, the distance $d(\overline{\boldsymbol{x}}, x^*)$ measures how far (in terms of revenues) the considered STGM-EP is from an ideal method that would *always* provide the optimal decisions. In applications, the value $d(\overline{\boldsymbol{x}}, x^*)$ represents the expected missing revenues that results from the implementation of any (randomly chosen) realization $\overline{x}$ of $\overline{\boldsymbol{x}}$.

It follows from Proposition 2.4 that a decision-maker will select the STGM-EP that provides the smallest value of $d(\overline{\boldsymbol{x}}, x^*)$. This selection criterion assesses the absolute quality of a method, i.e., in comparison to the ideal method. In applications, $Q(x^*)$ is typically not known, hence the decision-maker will rather compare the relative efficiency of two methods, as shown in Definition 2.5.

Definition 2.5 (Selection criterion). *Let A and B be two couples of scenario-tree generation method and extension procedure that yield random policies $\overline{\boldsymbol{x}}_A$ and $\overline{\boldsymbol{x}}_B$, respectively. We say that A is better than B for the stochastic programming problem if*

$$\mathbb{E}_{\overline{\boldsymbol{x}}_A}[Q(\overline{\boldsymbol{x}}_A)] > \mathbb{E}_{\overline{\boldsymbol{x}}_B}[Q(\overline{\boldsymbol{x}}_B)]. \tag{20}$$

Criterion (20) guarantees that the whole family of policies obtained by A is of better quality on average than those obtained by B. For this reason, we say that the selection criterion (20) is in the *average-case setting*. This setting is particularly relevant when the problem is solved regularly, each time with new data to infer the distribution of the random parameters. In that case, one runs the selection test just once, and uses the selected STGM-EP to obtain a new decision policy for each new data set. It is reasonable to expect the selected STGM-EP to perform well on the other data sets, provided of course the latter do not change the distribution too much, i.e., provided they affect the parameters of the distribution and not the very nature of the distribution itself. Moreover, the law of large numbers guarantees that the average performance of a method is a relevant quality criterion for a problem solved regularly.

It is worth noting an alternative selection setting that suits better decision-makers who intend to solve always the *exact* same problem. We call it a comparison in the *best-case setting* because it is based on the best policy $\overline{\boldsymbol{x}}(\omega)$, for all $\omega \in \Omega$, that a STGM-EP yields. In the best-case setting, A is better than B if

$$\sup_{\omega\in\Omega} Q(\overline{\boldsymbol{x}}_A(\omega)) > \sup_{\omega\in\Omega} Q(\overline{\boldsymbol{x}}_B(\omega)). \tag{21}$$

The interpretation of (21) is the following: if the decision-maker has unlimited computational resources and carries out A and B enough times, then *eventually* A will yield a policy with expected revenues higher than those obtained with any policy yielded by B. An estimator for $\sup_{\omega\in\Omega} Q(\overline{\boldsymbol{x}}(\omega))$ is $\max_{k=1,\dots,K} Q(\overline{x}^k)$, where $\overline{x}^1, \dots, \overline{x}^K$ are K realizations of $\overline{\boldsymbol{x}}$, which links the selection criterion (21) with the policy selection technique developed by Defourny et al. (2013). The drawback of (21) lies in the potentially long time required to find the best policy. This prevents the criterion to be used when the problem is solved regularly with different data, since the decision-maker will have to go through the selection test to find the best policy for each new data set.

The criterion (20) in the average-case setting can be slightly modified to assess not only the expected revenues, but also the stability of the STGM-EP with regard to its repeated use. We say that a method is *stable* if it yields decision policies that are close to each other in terms of expected revenues (otherwise it is called *unstable*). The importance of stability in the evaluation of scenario-tree generation methods has been discussed in Kaut and Wallace (2007). In our framework, a measure of stability is provided by the variance $\mathrm{Var}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$; the lower the variance, the more stable the method. A decision-maker may substitute in (20) the expectation $\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$ with $\delta\,\mathbb{E}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})] - (1-\delta)\mathrm{Var}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})]$, for $\delta \in (0,1)$, to include a measure of stability in the selection criterion. Obviously, deterministic methods are the most stable, since $\overline{\boldsymbol{x}}$ is not random and therefore $\mathrm{Var}_{\overline{\boldsymbol{x}}}[Q(\overline{\boldsymbol{x}})] = 0$.

3 Statistical estimation of the quality parameters

In this section, we show how to estimate the quality parameters introduced in Section 2. After introducing their statistical estimators in Section 3.1, we derive in Section 3.2 their confidence intervals, and in Section 3.3 we provide an algorithmic procedure to find the optimal sample sizes, defined as those minimizing the confidence interval bound for a given computational time.

3.1 Estimators

The quality parameters are estimated by sampling the probability distribution of $\boldsymbol{\xi}$ and the probability measure of $\widetilde{\boldsymbol{x}}$ (or $\overline{\boldsymbol{x}}$). Since the latter is sampled by generating several scenario trees (see Figure 1), and by solving the deterministic program (7)-(10) for each one, it is typically more costly to sample $\widetilde{\boldsymbol{x}}$ (or $\overline{\boldsymbol{x}}$) than $\boldsymbol{\xi}$. For this reason, it is relevant to consider an estimator that samples K times the random policy and $K \times M$ times the stochastic process, and to leave a degree of freedom in choosing how to balance the relative values of K and M to maximize the efficiency of the estimators.

We define the estimators of $p(t)$, CR, and $\mathbb{E}_{\boldsymbol{x}}[Q(\overline{\boldsymbol{x}})]$ as follows:

$$\widehat{p(t)}_{K,M} = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{M}\sum_{m=1}^{M}\mathbf{1}_{\widetilde{\Xi}_{..t}(\widetilde{x}^k)}(\xi_{..t}^{k,m}), \tag{22}$$

$$\widehat{\mathrm{CR}}_{K,M} = (\widehat{p(T)}_{K,M})^{-1}\left(\frac{1}{K}\sum_{k=1}^{K}\frac{1}{M}\sum_{m=1}^{M} q(\widetilde{x}^k(\xi^{k,m});\xi^{k,m})\mathbf{1}_{\widetilde{\Xi}_{..T}(\widetilde{x}^k)}(\xi^{k,m})\right), \tag{23}$$

$$\widehat{\mathbb{E}_{\boldsymbol{x}}[Q(\boldsymbol{x})]}_{K,M} = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{M}\sum_{m=1}^{M} q(\overline{x}^k(\xi^{k,m});\xi^{k,m}), \tag{24}$$

where $\{\widetilde{x}^k \,|\, k = 1,\dots,K\}$ and $\{\overline{x}^k \,|\, k = 1,\dots,K\}$ are two sets of K independent and identically distributed (i.i.d.) realizations of $\widetilde{\boldsymbol{x}}$ and $\overline{\boldsymbol{x}}$, respectively; $\{\xi^{k,m} \,|\, k = 1,\dots,K; m = 1,\dots,M\}$ is a set of $K \times M$ i.i.d. sample points of $\boldsymbol{\xi}$; $\xi_{..t}^{k,m}$ is the shorthand for $(\xi_1^{k,m},\dots,\xi_t^{k,m})$; and the notation $\mathbf{1}_U(\cdot)$, for some set U, is the indicator function:

$$\mathbf{1}_U(u) := \begin{cases} 1 & \text{if } u \in U; \\ 0 & \text{otherwise.} \end{cases} \tag{25}$$

We emphasize that each estimator (22)-(24) is computed using $K \times M$ out-of-sample scenarios.

3.2 Confidence interval

To derive a confidence interval for the quality parameters, it is convenient to introduce a single notation for them, and to do all the mathematical developments with it. To this end, we define the quantity of interest θ that we want to estimate:

$$\theta := \mathbb{E}[\phi(\boldsymbol{x},\boldsymbol{\xi})], \tag{26}$$

where $\phi : \left(\mathbb{R}^s \times \Pi_{t=1}^{T}\mathcal{L}_p(\Xi_{..t};\mathbb{R}^s)\right) \times \Xi \to \mathbb{R}$ is a map whose definition varies depending on the considered quality parameter, and $\boldsymbol{x}$ denotes a random policy being either $\widetilde{\boldsymbol{x}}$ or $\overline{\boldsymbol{x}}$. Throughout this section, for the sake of clarity, we do not add the subscript $(\boldsymbol{x},\boldsymbol{\xi})$ to the expectation, probability, variance, and covariance operators. If $\theta = p(t)$, we have

$$\phi(\widetilde{\boldsymbol{x}},\boldsymbol{\xi}) = \mathbf{1}_{\widetilde{\Xi}_{..t}(\widetilde{\boldsymbol{x}})}(\boldsymbol{\xi}_{..t}), \tag{27}$$

and if $\theta = \mathbb{E}_{\boldsymbol{x}}[Q(\overline{\boldsymbol{x}})]$,

$$\phi(\overline{\boldsymbol{x}}, \boldsymbol{\xi}) = q(\overline{\boldsymbol{x}}(\boldsymbol{\xi}); \boldsymbol{\xi}). \tag{28}$$

As for CR, it is the ratio of two expectations of the form (26). Following the definition of the estimators (22)-(24), we define the estimator $\widehat{\theta}_{K,M}$ of θ as follows:

$$\widehat{\theta}_{K,M} = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{M}\sum_{m=1}^{M}\phi(x^k, \xi^{k,m}), \tag{29}$$

where $\{x^k\}$ is a set of K i.i.d. realizations of $\boldsymbol{x}$, and the sample points $\{\xi^{k,m}\}$ are defined as above. It is immediate to see that $\widehat{\theta}_{K,M}$ is an unbiased and consistent estimator of θ. To derive a confidence interval for θ, we assume that the following condition holds:

C4. The random variable $\phi(\boldsymbol{x}, \boldsymbol{\xi})$ is square-integrable: $\mathbb{E}[\phi(\boldsymbol{x}, \boldsymbol{\xi})^2] < +\infty$.

The following proposition provides a confidence interval for θ; the notation $[a \pm b]$ is a shorthand for the interval $[a - b, a + b]$.

Proposition 3.1. *Assume condition C4 holds. Then, a 100(1 - α)% asymptotic confidence interval for θ is*

$$\mathcal{I}_{K,M}^{1-\alpha} = \left[\widehat{\theta}_{K,M} \pm z_{1-\alpha/2}\Big(\frac{\beta + \gamma(M-1)}{KM}\Big)^{1/2}\right], \tag{30}$$

where z_α denotes the α-level quantile of a standard normal distribution and β and γ are given by

$$\beta = \mathrm{Var}[\phi(\boldsymbol{x}, \boldsymbol{\xi})], \tag{31}$$

$$\gamma = \mathrm{Cov}[\phi(\boldsymbol{x}, \boldsymbol{\xi}^1), \phi(\boldsymbol{x}, \boldsymbol{\xi}^2)], \tag{32}$$

with $\boldsymbol{\xi}^1$ and $\boldsymbol{\xi}^2$ two i.i.d copies of $\boldsymbol{\xi}$.

Proof. Consider the random variables $U_k := \frac{1}{M}\sum_{m=1}^{M}\phi(x^k, \xi^{k,m})$ for all $k \in \{1, \ldots, K\}$. These random variables are i.i.d. because of the i.i.d. assumption on $\{x^k\}$ and $\{\xi^{k,m}\}$. We shall now verify that $\mathbb{E}[U_k^2] < \infty$ to apply the central limit theorem to $\frac{1}{K}\sum_{k=1}^{K}U_k$. We have

$$\mathbb{E}[U_k^2] = \frac{1}{M^2}\Big[\sum_{m=1}^{M}\mathbb{E}\big[\phi(x^k, \xi^{k,m})^2\big] + \sum_{m=1}^{M}\sum_{\substack{m'=1 \\ m' \neq m}}^{M}\mathbb{E}\big[\phi(x^k, \xi^{k,m})\,\phi(x^k, \xi^{k,m'})\big]\Big], \tag{33}$$

and by Cauchy-Schwarz inequality the expectation in the double sum is bounded by $\mathbb{E}[\phi(x^k, \xi^{k,m})^2]$. Therefore, Condition C4 implies that $\mathbb{E}[U_k^2]$ is finite for any k.

The central limit theorem applied to $\frac{1}{K}\sum_{k=1}^{K}U_k$ yields the following convergence:

$$\mathbb{P}\Big(\Big|\frac{K^{1/2}}{\mathrm{Var}[U_1]^{1/2}}\Big(\frac{1}{K}\sum_{k=1}^{K}U_k - \theta\Big)\Big| \leq z_{1-\alpha/2}\Big) \underset{K\to+\infty}{\longrightarrow} \mathbb{P}(|Z| \leq z_{1-\alpha/2}) = 1 - \alpha, \tag{34}$$

where Z follows a standard normal distribution and z_α denotes the α-level quantile of Z. Thus, a 100(1 - α)% asymptotic confidence interval for θ is

$$\mathcal{I}_{K,M}^{1-\alpha} = \left[\widehat{\theta}_{K,M} \pm z_{1-\alpha/2}\frac{\mathrm{Var}[U_1]^{1/2}}{K^{1/2}}\right]. \tag{35}$$

The quantity $\mathrm{Var}[U_1]$ can be simplified using the fact that the random variables $\phi(x^1, \xi^{1,m})$, for all $m \in \{1, \ldots, M\}$, are i.i.d:

$$\mathrm{Var}[U_1] = \frac{1}{M^2} \sum_{m=1}^{M} \mathrm{Var}[\phi(x^1, \xi^{1,m})] + \frac{1}{M^2} \sum_{m=1}^{M} \sum_{\substack{m'=1 \\ m' \neq m}}^{M} \mathrm{Cov}[\phi(x^1, \xi^{1,m}), \phi(x^1, \xi^{1,m'})] \tag{36}$$

$$= \frac{1}{M} \mathrm{Var}[\phi(x^1, \xi^{1,m})] + \frac{M-1}{M} \mathrm{Cov}[\phi(x^1, \xi^{1,1}), \phi(x^1, \xi^{1,2})]. \tag{37}$$

Finally, defining $\beta = \mathrm{Var}[\phi(x^1, \xi^{1,m})]$ and $\gamma = \mathrm{Cov}[\phi(x^1, \xi^{1,1}), \phi(x^1, \xi^{1,2})]$, and combining (35) and (37), yields the confidence interval

$$\mathcal{I}_{K,M}^{1-\alpha} = \left[\widehat{\theta}_{K,M} \pm z_{1-\alpha/2} \Big(\frac{\beta + \gamma(M-1)}{KM} \Big)^{1/2} \right]. \tag{38}$$

□

The quantities β and γ, defined in (31) and (32), are not available analytically for the same reason as θ. They can be estimated through the following consistent and unbiased estimators $\widehat{\beta}_{K,M}$ and $\widehat{\gamma}_{K,M}$ that use the same sample sets $\{x^k\}$ and $\{\xi^{k,m}\}$ as $\widehat{\theta}_{K,M}$:

$$\widehat{\beta}_{K,M} = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{M} \sum_{m=1}^{M} \phi(x^k, \xi^{k,m})^2 - (\widehat{\theta}_{K,M})^2, \tag{39}$$

$$\widehat{\gamma}_{K,M} = \frac{1}{K} \sum_{k=1}^{K} \Big(\frac{1}{M} \sum_{m=1}^{M} \phi(x^k, \xi^{k,m}) \Big)^2 - (\widehat{\theta}_{K,M})^2. \tag{40}$$

Finally, let us observe that a deterministic STGM-EP always satisfies $\gamma = 0$. Indeed, a deterministic STGM-EP yields a nonrandom policy (i.e., $\boldsymbol{x}(\omega) = x$ for all $\omega \in \Omega$), therefore $\gamma = \mathrm{Cov}[\phi(x, \boldsymbol{\xi}^1), \phi(x, \boldsymbol{\xi}^2)] = 0$, because $\boldsymbol{\xi}^1$ and $\boldsymbol{\xi}^2$ are independent. As a result, when applied to a deterministic STGM-EP, the estimator and the confidence interval will be set with $K = 1$ and $\gamma = 0$.

3.3 Optimal sample size selection

The bound of the confidence interval (30) depends on the two sample sizes K and M. There is a degree of freedom in choosing how to balance the values of K and M, and this choice will clearly affect the estimation quality of θ. We propose a systematic procedure to find K and M in an optimal way, i.e., to minimize the bound of the confidence interval for a given computational time.

We denote by t_0 and t_1 the times required to obtain one realization x^k of $\boldsymbol{x}$ and $\xi^{k,m}$ of $\boldsymbol{\xi}$, respectively, and by t_2 the time required to compute the quantity $\phi(x^k, \xi^{k,m})$. Thus, the whole computation of $\widehat{\theta}_{K,M}$ takes $Kt_0 + KM(t_1 + t_2)$ units of time. We denote by $\tau > 0$ the total computational time available to the decision-maker (of course it is required that $\tau \geq t_0 + t_1 + t_2$).

The optimal sample sizes K^* and M^* that minimize the confidence interval bound for a computational time τ are the optimal solutions of the following program:

$$P_\tau(\beta, \gamma): \quad \min_{K,M} \quad \frac{\beta + \gamma(M-1)}{KM} \tag{41}$$

$$s.t. \quad Kt_0 + KM(t_1 + t_2) \leq \tau, \tag{42}$$

$$K \in \mathbb{N}^*, M \in \mathbb{N}^*. \tag{43}$$

It is also possible to consider the reverse problem, i.e., to find K^* and M^* that minimize the computational time required to have the value $\frac{\beta+\gamma(M-1)}{KM}$ lower than some target $v > 0$. In that case, K^* and M^* are the optimal solutions of the following program:

$$P_v(\beta,\gamma): \quad \min_{K,M} \quad Kt_0 + KM(t_1+t_2) \tag{44}$$

$$s.t. \quad \frac{\beta+\gamma(M-1)}{KM} \leq v, \tag{45}$$

$$K \in \mathbb{N}^*, M \in \mathbb{N}^*. \tag{46}$$

We describe now the procedure that we propose to estimate the quality parameters of a stochastic STGM-EP:

(i) compute the estimators $\widehat{\beta}_{K_0,K_0}$ and $\widehat{\gamma}_{K_0,K_0}$ for two values $K_0 \geq 2$ and $M_0 \geq 2$ (set empirically to have a fast but fairly accurate estimation), and estimate t_0, t_1, and t_2;

(ii) solve the program $P_\tau(\widehat{\beta}_{K_0,M_0}, \widehat{\gamma}_{K_0,M_0})$ for the time limit τ, and retrieve (K^*, M^*);

(iii) compute $\widehat{\theta}_{K^*,M^*}$, $\widehat{\beta}_{K^*,M^*}$, $\widehat{\gamma}_{K^*,M^*}$, and derive the confidence interval $\mathcal{I}^{1-\alpha}_{K^*,M^*}$.

A decision-maker interested in having highly reliable estimates, regardless of the computational cost, can substitute in step (ii) the program $P_\tau(\widehat{\beta}_{K_0,M_0}, \widehat{\gamma}_{K_0,M_0})$ with $P_v(\widehat{\beta}_{K_0,M_0}, \widehat{\gamma}_{K_0,M_0})$ for some variance target $v > 0$.

As for the estimation of the quality parameters of a deterministic STGM-EP, it is done by setting $K = 1$ and by letting M be as large as possible within the computational time limit.

In Section 5.4 of the numerical experiments, we will prove the relevance of the optimal sample size selection by comparing the efficiency of the estimator (29) with a more classical estimator that samples $\boldsymbol{x}$ and $\boldsymbol{\xi}$ together.

4 Proposed procedures to extend the tree policy

4.1 Nearest-neighbor extension

The nearest-neighbor (NN) extension assigns to $\widetilde{x}_t(\xi_{..t})$ the value of the decisions $\widehat{x}^{n*}$ at the node $n \in \mathcal{N}_t$ nearest to $\xi_{..t}$. Several nearest-neighbor extensions can be defined depending on (i) the metric used to define the distance between $\xi_{..t}$ and $\zeta^{\cdot n}$, and (ii) the subset of $\mathcal{N}_t$ in which the nearest node is searched. For (i), we choose the stage-t Euclidean metric $\|\cdot\|_t$ defined as

$$\|u\|_t = \Big(\sum_{i=1}^{t}\sum_{j=1}^{d}|(u_i)_j|^2\Big)^{1/2}, \tag{47}$$

where $u = (u_1, \ldots, u_t) \in \mathbb{R}^d \times \cdots \times \mathbb{R}^d$ and $(u_i)_j$ denotes the j-th component of u_i; for convenience, we also denote $\|u_i\|^2 = \sum_{j=1}^{d}|(u_i)_j|^2$. For (ii), two relevant choices exist. The first is to search the nearest node among the whole stage-t nodes $\mathcal{N}_t$; the second is to search only among the children nodes $C(m)$, where $m \in \mathcal{N}_{t-1}$ is the node corresponding to the decisions made at stage $t-1$. We refer to the first choice as NN-AT (AT standing for *across tree*) and to the second as NN-AC (*across children*). We note that both extensions yield the same stage-1 decision function, therefore, they coincide for two-stage problems.

Nearest-neighbor extension across tree (NN-AT)

The extension *across tree* searches for the nearest node among the whole of $\mathcal{N}_t$. It allows to switch among the branches of the scenario tree, hence it is less sensitive than NN-AC to the decisions made at the previous stages. The node n nearest to the realization $\xi_{..t}$ is chosen by comparing the whole history of the random parameters up to stage t, by means of the norm (47). Thus, the set $\Xi_{..t}$ is partitioned into $|\mathcal{N}_t|$ Voronoï cells $V^n_{t,\mathrm{AT}}$ defined as

$$V^n_{t,\mathrm{AT}} = \big\{\xi_{..t} \in \Xi_{..t} \;\big|\; \forall r \in \mathcal{N}_t \setminus \{n\},\ \|\xi_{..t} - \zeta^{..n}\|_t < \|\xi_{..t} - \zeta^{..r}\|_t\big\}, \tag{48}$$

for all $n \in \mathcal{N}_t$. On each cell $V^n_{t,\mathrm{AT}}$, the decision function is constant and yields the decision $\widehat{x}^{n*}$. Thus, the stage-t decision function is piecewise constant on $\Xi_{..t}$ and takes the form

$$\widetilde{x}^{\mathrm{AT}}_t(\xi_{..t}) = \sum_{n \in \mathcal{N}_t} \widehat{x}^{n*}\, \mathbf{1}_{V^n_{t,\mathrm{AT}}}(\xi_{..t}), \quad \forall \xi_{..t} \in \Xi_{..t}, \tag{49}$$

where $\mathbf{1}_{V^n_{t,\mathrm{AT}}}(\cdot)$ is the indicator function (defined in (25)).

Nearest-neighbor extension across children (NN-AC)

The extension *across children* searches for the nearest node at stage t among the set $C(m)$, where $m \in \mathcal{N}_{t-1}$ is the node of the decisions at the previous stage. This extension is computationally advantageous because it does not require to partition the whole set $\Xi_{..t}$ at every stage, but only its subset $V^m_{t,\mathrm{AC}} \times \Xi_t$ with $V^m_{t,\mathrm{AC}}$ the Voronoï cell of m. This set is partitioned into $|C(m)|$ Voronoï cells $V^n_{t,\mathrm{AC}}$ defined as

$$V^n_{t,\mathrm{AC}} = V^m_{t,\mathrm{AC}} \times \big\{\xi_t \in \Xi_t \;\big|\; \forall r \in C(m) \setminus \{n\},\ \|\xi_t - \zeta^n\| < \|\xi_t - \zeta^r\|\big\}, \tag{50}$$

for all $m \in C(n)$. The stage-t decision function is piecewise constant on $\Xi_{..t}$ and takes the form

$$\widetilde{x}^{\mathrm{AC}}_t(\xi_{..t}) = \sum_{n \in C(m)} \widehat{x}^{n*}\, \mathbf{1}_{V^n_{t,\mathrm{AC}}}(\xi_{..t}), \quad \forall \xi_{..t} \in V^m_{t,\mathrm{AC}} \times \Xi_t. \tag{51}$$

We illustrate the stage-1 decision function of the nearest-neighbor extension in Figure 2 (a), and the Voronoï cells in Figure 3 (a)-(b); they correspond to the scenario tree represented in Figure 4.

4.2 N-nearest-neigbhor-weighted extension (NNNW)

The nearest-neighbor extension can be generalized to a weighted average over the N-*nearest* nodes (denoted by NNNW). To define it formally, let us denote by $\mathcal{V}_N(\xi_{..t}) \subseteq \mathcal{N}_t$, for $N \geq 2$, the set of the N-nearest stage-t nodes to $\xi_{..t}$ for the metric (47). The stage-t decision function of the NNNW extension is given by

$$\widetilde{x}_t(\xi_{..t}) = \sum_{n \in \mathcal{V}_N(\xi_{..t})} \lambda^n(\xi_{..t})\, \widehat{x}^{n*}, \quad \forall \xi_{..t} \in \Xi_{..t}, \tag{52}$$

where $\lambda^n(\cdot)$ is a weight function that we define as

$$\lambda^n(\xi_{..t}) = \Bigg[\sum_{l \in \mathcal{V}_N(\xi_{..t})} \prod_{m \in \mathcal{V}_N(\xi_{..t}) \setminus \{l\}} \|\xi_{..t} - \zeta^{..m}\|_t\Bigg]^{-1} \prod_{m \in \mathcal{V}_N(\xi_{..t}) \setminus \{n\}} \|\xi_{..t} - \zeta^{..m}\|_t, \tag{53}$$

for every $n \in \mathcal{V}_N(\xi_{..t})$. This definition is justified by the fact that $\lambda^n(\cdot)$ satisfies:

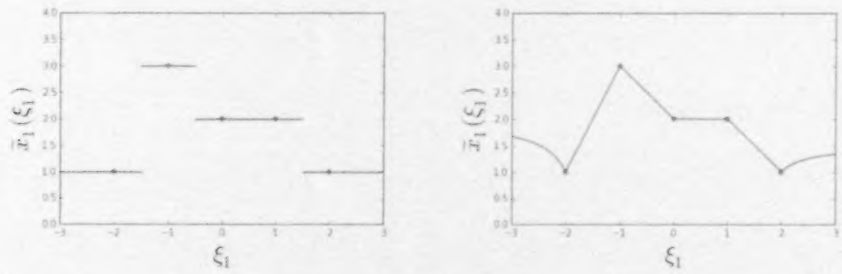
(i) $\lambda^n(\cdot) \geq 0$;

(ii) $\sum_{n \in \mathcal{V}_N(\xi_{\cdot t})} \lambda^n(\xi_{\cdot t}) = 1$ for every $\xi_{\cdot t} \in \Xi_{\cdot t}$;

(iii) $\lambda^n(\zeta^{\cdot n}) = 1$ and $\lambda^m(\zeta^{\cdot n}) = 0$ for every $m \in \mathcal{V}_N(\zeta^{\cdot n}) \setminus \{n\}$.

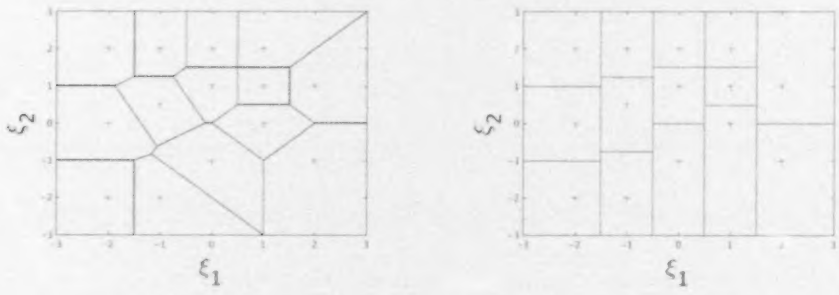
The properties (i) and (ii) imply that $\widetilde{x}_t(\xi_{\cdot t})$ is a convex combination of the tree decisions $\widehat{x}^{n*}$ for $n \in \mathcal{V}_N(\xi_{\cdot t})$, and (iii) ensures that $\widetilde{x}_t(\cdot)$ satisfies $\widetilde{x}_t(\zeta^{\cdot n}) = \widehat{x}^{n*}$, which is a requirement of Definition 1.1. We note that since $\widetilde{x}_t(\xi_{\cdot t})$ is a convex combination of $\widehat{x}^{n*}$, it may fail to provide integer values even if all $\widehat{x}^{n*}$ are integers. For this reason, the $NNNW$ extension cannot be used directly for integer programs, unless some techniques are introduced to restrict $\widetilde{x}_t(\cdot)$ to a set of integers.

An illustration of this extension is displayed in Figure 2 (b), for the scenario tree in Figure 4.



(a) NN (b) 2NNW

Figure 2: First-stage extended decision functions of NN (a) and 2NNW (b), for the scenario tree in Figure 4.



(a) NN-AT (b) NN-AC

Figure 3: Voronoï cells (48) and (50) in the support of $(\boldsymbol{\xi}_1, \boldsymbol{\xi}_2)$ for NN-AT (a) and NN-AC (b), and for the scenario tree in Figure 4. The +-markers are the sites of the cells.

5 Numerical experiments

5.1 Preliminaries of the numerical experiments

In this section, we apply the quality evaluation framework developed in Section 2, 3, and 4 on two case studies: a two-stage newsvendor problem and a four-stage multi-product assembly problem; for the latter, we use the same data as in Defourny et al. (2013). We generate the scenario trees by

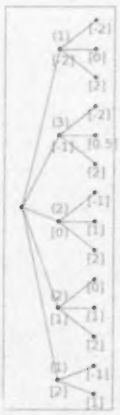


Figure 4: An example of a 3-stage scenario tree ($T = 2$). The values in bracket are the discretization points ζ^n for $n \in \mathcal{N}_1 \cup \mathcal{N}_2$. The values in parenthesis are the optimal decisions $\widehat{x}^{n*}$ for $n \in \mathcal{N}_1$ (only shown at stage 1).

three different methods: optimal quantization (OQ), randomized quasi-Monte Carlo (RQMC), and Monte Carlo (MC) (see the corresponding references in Section 1.2). The tree structures are chosen beforehand with constant branching coefficients, i.e., $|C(n)|$ is constant for all $n \in \mathcal{N} \setminus \mathcal{N}_T$. The tree decisions are extended by the three extension procedures introduced in Section 4: nearest-neigbhor across tree (NN-AT), nearest-neigbhor across children (NN-AC), and two-nearest-neigbhor-weighted (2NNW). The resulting STGM-EPs (summarized in Table 1) are compared by means of the quality parameters, and the selection of the best method is done in the average-case setting.

The generation of scenario trees for both case studies is based on the discretization of a standard normal $\mathcal{N}(0,1)$ distribution. Discretization by the OQ method is done by Algorithm 2 in Pflug and Pichler (2015), which minimizes the Wasserstein distance of order 2 between the $\mathcal{N}(0,1)$ distribution and its approximation sitting on finitely many points. This method provides a set of discretization points along with the corresponding probabilities.

Discretization by the RQMC method is done by the technique of *randomly shifted lattice rules* (see, e.g, L'Ecuyer and Lemieux (2000) and Sloan et al. (2002)). This technique randomizes a low discrepancy set of N points in $[0,1]$ and transforms it with the inverse $\mathcal{N}(0,1)$ cumulative $\phi^{-1} : (0,1) \to \mathbb{R}$. The output set of points is

$$\left\{\phi^{-1}(\{i/N + u\}) \,|\, i = 0, \ldots, N-1\right\}, \tag{54}$$

where u is a realization of a uniform distribution in $[0,1]$ and $\{\cdot\}$ is the fractional part function. The weight corresponding to each point is set to $1/N$, as it is customary in quasi-Monte Carlo. The set of points (54) enjoys two important properties, making it interesting for discretization the $\mathcal{N}(0,1)$ distribution: (i) each point has a marginal $\mathcal{N}(0,1)$ distribution, (ii) the points are not independent of each other, they cover uniformly the part of the support of $\mathcal{N}(0,1)$ where the probability mass is concentrated. We note that each new realization u implies a new set of points, therefore, RQMC is a *stochastic* scenario-tree generation method. We refer for instance to Koivu (2005) for the use of the RQMC method in stochastic programming.

Discretization by the MC method provides the so-called *Sample Average Approximation* (see Shapiro (2003)). Although MC is known by theoreticians to be less efficient than RQMC and OQ for sampling in small dimension, it is often used by practitioners because it remains the most natural and easiest way to generate scenarios. For this reason, we include it in the evaluation test.

The numerical experiments are implemented in Python 2.7.4 on a Linux machine with Intel Xeon X5472 @ 3.00GHz. We use CPLEX 12.6.1.0 with default setting to solve the scenario-tree deterministic programs.

Notation	Description
NN-AC	nearest-neighbor extension across children
NN-AT	nearest-neighbor extension across tree
2NNW	2-nearest-neighbor-weighted extension across tree

(a) Extension procedures

Notation	Description
OQ	optimal quantization method
RQMC	randomized quasi-Monte Carlo method
MC	Monte Carlo method

(b) Scenario-tree generation methods

Table 1: STGM-EPs considered in the numerical experiments.

5.2 Case study 1: the newsvendor problem

The newsvendor problem is stated as follows: A newsvendor buys to a supplier x_0 newspapers at stage 0 at a fixed price a. At stage 1, the newsvendor sells $x_{1,1}$ newspapers at price b and returns $x_{1,2}$ to the supplier, the latter pays c for each newspaper returned. Demand for newspapers is given by a positive random variable $\boldsymbol{\xi}_1$. Although the decisions involve integer variables, it is customary to relax the integrity constraints and to consider the corresponding continuous problem (see, e.g., Birge and Louveaux (1997)). The two-stage stochastic program takes the form

$$\max_{(x_0, x_{1,1}, x_{1,2})} \quad -a\,x_0 + \mathbb{E}[b\,x_{1,1}(\boldsymbol{\xi}_1) + c\,x_{1,2}(\boldsymbol{\xi}_1)] \tag{55}$$

$$\text{s.t.} \quad x_{1,1}(\boldsymbol{\xi}_1) \leq \boldsymbol{\xi}_1; \tag{56}$$

$$x_{1,1}(\boldsymbol{\xi}_1) + x_{1,2}(\boldsymbol{\xi}_1) \leq x_0; \tag{57}$$

$$x_0 \in \mathbb{R}_+,\ x_{1,1}(\boldsymbol{\xi}_1) \in \mathbb{R}_+,\ x_{1,2}(\boldsymbol{\xi}_1) \in \mathbb{R}_+. \tag{58}$$

The parameters are set to $a = 2$, $b = 5$, and $c = 1$. The demand $\boldsymbol{\xi}_1$ follows a log-normal distribution, i.e., $\log(\boldsymbol{\xi}_1)$ follows a $\mathcal{N}(\mu, \sigma^2)$ distribution, the mean is set to $\mu = \log(200)$ and the variance to $\sigma^2 = 1/2$ (these values for the parameters are taken from Proulx (2014)).

The optimal value of (55)-(58) rounded off to the second decimal is $Q(x^*) = 500.25$. The optimal stage-1 decision functions are $x^*_{1,1}(\boldsymbol{\xi}_1) = \min(x^*_0, \boldsymbol{\xi}_1)$ and $x^*_{1,2}(\boldsymbol{\xi}_1) = \max(x^*_0 - \boldsymbol{\xi}_1, 0)$.

The numerical experiments are performed with the methods in Table 1 and for scenario trees with 5, 20, 40, and 80 scenarios. Although the resulting scenario trees have small sizes, we will see that clear conclusions can be drawn from them. Moreover, we wish to assess the quality of methods with small scenario sizes, because a method that performs well with few scenarios can also be used to solve a generalization of the problem with more stages. Indeed, having good quality decisions with only 5 scenarios opens the door to the solution of the problem with, e.g., 10 stages, since a 10-stage scenario tree with a branching of 5 nodes at every stage remains tractable ($5^{10-1} \simeq 2 \times 10^6$ nodes). However, if 80 scenarios are required to obtain good quality decisions for the problem with 2 stages, then the same problem extended to 10 stages is intractable for the tested method ($80^{10-1} \simeq 10^{17}$ nodes).

Optimal selection of sample sizes:

The quality parameters are estimated using the procedure described in Section 3.3, for a computational time limit of one hour for each STGM-EP. The optimal sample sizes K^* and M^* are displayed in Table 2 (since the extensions *across tree* or *across children* coincide for two-stage problems, we remove the suffix -AT and -AC).

	5 scen.		20 scen.		40 scen.		80 scen.	
	K^*	M^*	K^*	M^*	K^*	M^*	K^*	M^*
OQ-NN	1	19×10^6	1	9×10^6	1	5×10^6	1	3×10^6
OQ-2NNW	1	10×10^6	1	6×10^6	1	4×10^6	1	2×10^6
RQMC-NN	168×10^3	32	64×10^3	50	46×10^3	19	20×10^3	36
RQMC-2NNW	161×10^3	19	66×10^3	28	43×10^3	21	20×10^3	32
MC-NN	179×10^3	25	79×10^3	24	39×10^3	38	21×10^3	28
MC-2NNW	152×10^3	23	62×10^3	34	33×10^3	46	18×10^3	41

Table 2: Optimal sample sizes K^* and M^* for a limit of 1h of computation for each STGM-EP. The values M^* are rounded-off to the nearest 10^6 for OQ; the values K^* are rounded-off to the nearest 10^3 for RQMC and MC.

We have that $K^* = 1$ for the optimal quantization method, since it is a deterministic way to generate scenario trees (see Remark 1.1 and the discussion at the end of Section 3.2). We emphasize that each value $K^* \times M^*$ is the number of out-of-sample scenarios used to test the corresponding STGM-EP.

Quality parameters $p(1)$ and CR:

The probability of feasibility $p(1)$ and the conditional revenues CR given the whole feasibility of the extended tree policy $\widetilde{\boldsymbol{x}} = (\widetilde{\boldsymbol{x}}_0, \widetilde{\boldsymbol{x}}_{1,1}, \widetilde{\boldsymbol{x}}_{1,2})$ take the form (see Definition 2.1):

$$p(1) = \mathbb{P}_{(\boldsymbol{\xi}_1, \widetilde{\boldsymbol{x}})}[\boldsymbol{\xi}_1 \in \widetilde{\Xi}_1(\widetilde{\boldsymbol{x}})], \tag{59}$$

$$\mathrm{CR} = \mathbb{E}_{(\boldsymbol{\xi}_1, \widetilde{\boldsymbol{x}})}[-a\,\widetilde{\boldsymbol{x}}_0 + b\,\widetilde{\boldsymbol{x}}_{1,1}(\boldsymbol{\xi}_1) + c\,\widetilde{\boldsymbol{x}}_{1,2}(\boldsymbol{\xi}_1) \,|\, \boldsymbol{\xi}_1 \in \widetilde{\Xi}_1(\widetilde{\boldsymbol{x}})], \tag{60}$$

where $\widetilde{\Xi}_1(\widetilde{x}) = \{\xi_1 \in \Xi_1 \,|\, (\widetilde{x}_0, \widetilde{x}_{1,1}(\xi_1), \widetilde{x}_{1,2}(\xi_1))$ satisfy (56), (57), (58)$\}$. The estimates of $p(1)$ and CR are displayed in Table 3. To facilitate the comparison with the optimal value $Q(x^*)$, the estimates of CR are given in percentage of $Q(x^*)$ and are denoted in the table by $\mathrm{CR}_{\%}$.

It follows from the estimates in Table 3 that a clear hierarchy exists among the three scenario-tree generation methods: OQ is better than RQMC which, in turn, are better than MC, as well as between the two extension procedures: 2NNW is better than NN. This hierarchy can be schematized as

$$\mathrm{OQ} > \mathrm{RQMC} > \mathrm{MC} \quad \text{and} \quad \mathrm{2NNW} > \mathrm{NN}. \tag{61}$$

The couple OQ-2NNW with only 20 scenarios yields an extended tree policy that is very close to the optimal policy in terms of feasibility (99.8% of the time) and expected revenues (100.2% of the optimal ones); we recall that $\mathrm{CR}_{\%}$ may be greater than 100% when $p(1) < 1$, because $\mathrm{CR}_{\%}$ computes the expected revenues on a subset of values of the random parameters. This nearly achieves the goal we formulated in Section 1.3, i.e, to introduce the extension procedure to recover a decision policy of the original stochastic program, and to find the STGM-EP providing the closest policy to x^*.

	5 scen.		20 scen.		40 scen.		80 scen.	
	$p(1)$	$\mathrm{CR}_{\%}$	$p(1)$	$\mathrm{CR}_{\%}$	$p(1)$	$\mathrm{CR}_{\%}$	$p(1)$	$\mathrm{CR}_{\%}$
OQ-NN	0.618	102.1	0.620	114.4	0.622	116.8	0.623	118.2
OQ-2NNW	0.957	101.8	**0.998**	**100.2**	**0.999**	**100.1**	**0.997**	**100.3**
RQMC-NN	0.613	112.8	0.602	118.7	0.612	119.1	0.617	119.1
RQMC-2NNW	0.895	109.1	0.961	104.8	0.977	103.0	**0.985**	**102.0**
MC-NN	0.610	100.1	0.612	113.5	0.616	116.4	0.619	117.8
MC-2NNW	0.757	101.9	0.790	106.3	0.798	107.3	0.804	107.5

Table 3: Estimates of the quality parameters $p(1)$ and $\mathrm{CR}_{\%}$. Data in bold font single out the STGM-EPs that satisfy $p(1) \geq 0.98$ and $\mathrm{CR} \geq 99\% \times Q(x^*)$, which can be considered as satisfactory. Confidence intervals are not displayed for the sake of clarity; the widest 95% confidence interval for each column from left to right are: ± 0.0009; ± 0.11; ± 0.001; ± 0.2; ± 0.0014; ± 0.3; ± 0.0017; ± 0.3.

The second best couple, namely RQMC-2NNW, provides good quality decisions from 80 scenarios. However, even for this many scenarios, the probability of feasibility is statistically significantly below that of OQ-2NNW. Any other couple (including that made by combining MC and 2NNW) yields decisions with low probability of feasibility (less than 80%). In that case, the conditional revenues CR cannot be trusted, as we explained in the discussion following Definition 2.1.

Quality parameter $\mathbb{E}_{\boldsymbol{x}}[Q(\overline{\boldsymbol{x}})]$:

Since the stage-1 recourse decisions are known analytically for the newsvendor problem, a decision-maker is interested in the stage-0 decision only. For this reason, we assess the quality of the feasible extended tree policy $\overline{\boldsymbol{x}} = (\widetilde{\boldsymbol{x}}_0, r_{1,1}, r_{1,2})$, where $(r_{1,1}, r_{1,2})$ are the recourse decisions given by

$$r_{1,1}(x_0; \xi_1) = \min(x_0, \xi_1), \tag{62}$$

$$r_{1,2}(x_0; \xi_1) = \max(x_0 - \xi_1, 0). \tag{63}$$

The quality of $\overline{\boldsymbol{x}}$ is assessed through the expected revenues that it yields, i.e., through the third quality parameter given by

$$\mathbb{E}_{\boldsymbol{x}}[Q(\overline{\boldsymbol{x}})] = \mathbb{E}_{(\boldsymbol{\xi}_1, \widetilde{\boldsymbol{x}}_0)}[-a\,\widetilde{\boldsymbol{x}}_0 + b\,r_{1,1}(\widetilde{\boldsymbol{x}}_0; \boldsymbol{\xi}_1) + c\,r_{1,2}(\widetilde{\boldsymbol{x}}_0; \boldsymbol{\xi}_1)]. \tag{64}$$

Table 4 displays the estimates of $\mathbb{E}_{\boldsymbol{x}}[Q(\overline{\boldsymbol{x}})]$ in percentage of $Q(x^*)$, along with the corresponding 95% confidence interval bounds (column $\pm \mathrm{CI}_{95\%}$). Since only the stage-0 decision is assessed by (64), the estimates do not depend on an extension procedure.

	5 scen.		20 scen.		40 scen.		80 scen.	
	Est.	$\pm \mathrm{CI}_{95\%}$	Est.	$\pm \mathrm{CI}_{95\%}$	Est.	$\pm \mathrm{CI}_{95\%}$	Est.	$\pm \mathrm{CI}_{95\%}$
OQ	99.80	0.04	99.96	0.05	99.95	0.07	100.01	0.09
RQMC	98.71	0.08	99.78	0.12	100.00	0.22	99.92	0.25
MC	91.44	0.11	97.22	0.15	98.62	0.17	99.35	0.27

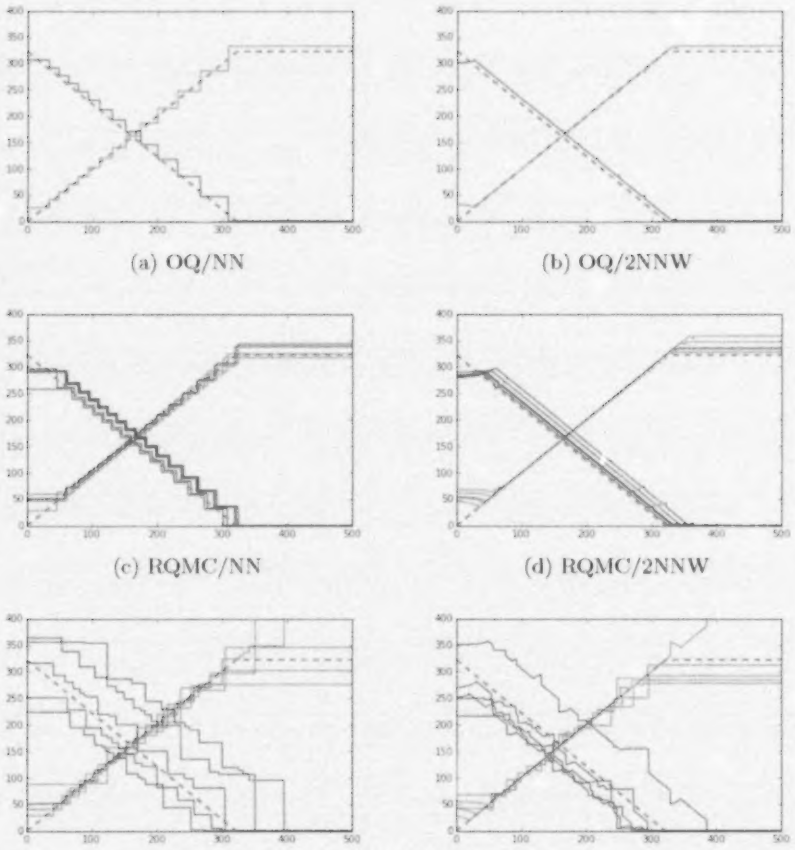
Table 4: Estimates of $\mathbb{E}_{\boldsymbol{x}}[Q(\overline{\boldsymbol{x}})]$ with $\overline{\boldsymbol{x}} = (\widetilde{\boldsymbol{x}}_0, r_{1,1}, r_{1,2})$ (given in percentage of $Q(x^*)$).

The OQ method achieves 99.8% of the optimal expected revenues with *only 5 scenarios*, while 20 scenarios are needed for the RQMC method to reach this value. From 40 scenarios, the distinction between OQ and RQMC is made impossible by the statistical error (though small: less than $\pm 0.25\%$), and both methods yield a decision close to optimality. The MC method yields the lowest quality stage-0 decision, which is statistically significantly below the OQ and RQMC methods for all tested scenario sizes. However, as far as the stage-0 decision is concerned, the use of MC with 80 scenarios is acceptable since the expected revenues are greater than 99% of the optimal ones. This last observation, combined with the estimates of $p(1)$ in Table 3, shows that the drawback of MC lies mostly in the poor quality of its stage-1 decision functions. This statement is supported by the graphical analysis done in the next paragraph.

Plots of the stage-1 decision functions

The extended stage-1 decision functions $\widetilde{x}_{1,1}$ and $\widetilde{x}_{1,2}$ are displayed for each STGM-EP in Figure 5, and are compared with their optimal counterparts $x^*_{1,1}$ and $x^*_{1,2}$. This graphical comparison allows to understand why some STGM-EPs provide expected revenues higher than others.

As expected from the above quantitative analysis, the couple OQ-2NNW in (b) yields stage-1 decisions that fit very well the optimal ones. The approximation quality is also quite good for RQMC-2NNW in (d). As for MC, in (e) and (f), it appears that the functions have a large variability due to the absence of a variance reduction technique, and an erratic behavior due to a discretization scheme that covers the support of $\boldsymbol{\xi}_1$ less uniformly than RQMC and OQ.



(a) OQ/NN (b) OQ/2NNW

(c) RQMC/NN (d) RQMC/2NNW

(e) MC/NN (f) MC/2NNW

Figure 5: Plots of $\widetilde{\boldsymbol{x}}_{1,1}$ and $\widetilde{\boldsymbol{x}}_{1,2}$ (solid lines) compared with $x^*_{1,1}$ and $x^*_{1,2}$ (dashed lines) for 20 scenarios. For RQMC and MC, five realizations of $\widetilde{\boldsymbol{x}}_{1,1}$ and $\widetilde{\boldsymbol{x}}_{1,2}$ are displayed on the same figure. The x-axis is the demand ξ_1.

5.3 Case study 2: the multi-product assembly problem

We consider the four-stage multi-product assembly problem presented in Defourny et al. (2013), Section 4.1, with the same numerical parameters and stochastic process. The stochastic program takes the form:

$$\max_{(x_0,\dots,x_3)} \quad -c_0^\top x_0 + \mathbb{E}\big[-c_1^\top x_1(\boldsymbol{\xi}_1) - c_2^\top x_2(\boldsymbol{\xi}_{..2}) + c_3^\top x_3(\boldsymbol{\xi}_{..3})\big] \tag{65}$$

$$\text{s.t.} \quad \sum_{j=1}^{8} A_{i,j}\, x_{1,j}(\boldsymbol{\xi}_1) \le x_{0,i}, \quad \forall i \in \{1,\dots,12\}; \tag{66}$$

$$\sum_{k=1}^{5} B_{j,k}\, x_{2,k}(\boldsymbol{\xi}_{..2}) \le x_{1,j}(\boldsymbol{\xi}_1), \quad \forall j \in \{1,\dots,8\}; \tag{67}$$

$$x_{3,k}(\boldsymbol{\xi}_{..3}) \le \max(0, b_{k,0} + \sum_{t=1}^{3} b_{k,t}\boldsymbol{\xi}_t), \quad \forall k \in \{1,\dots,5\}; \tag{68}$$

$$x_{3,k}(\boldsymbol{\xi}_{..3}) \le x_{2,k}(\boldsymbol{\xi}_{..2}), \quad \forall k \in \{1,\dots,5\}; \tag{69}$$

$$x_0 \in \mathbb{R}^{12}_+,\ x_1(\boldsymbol{\xi}_1) \in \mathbb{R}^{8}_+,\ x_2(\boldsymbol{\xi}_{..2}) \in \mathbb{R}^{5}_+,\ x_3(\boldsymbol{\xi}_{..3}) \in \mathbb{R}^{5}_+. \tag{70}$$

The interpretation of this problem is as follows: At stage 0, a quantity $x_{0,i}$ of a product i is purchased at a price $c_{0,i}$ per unit. At stage 1, a quantity $x_{1,j}$ of a product j is produced at a cost $c_{1,j}$ per unit from the quantity of products available at the previous stage, and in proportions given by the matrix A in (66). The same production scheme happens again at stage 2, with the quantity $x_{2,k}$, the cost $c_{2,k}$, and in proportions given by the matrix B in (67). At stage 3, a quantity $x_{3,k}$ of the final product k is sold at price $c_{3,k}$ per unit, and the demand for k is $\max(0, b_{k,0} + \sum_{t=1}^{3} b_{k,t}\boldsymbol{\xi}_t)$ where $\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \boldsymbol{\xi}_3$ are three i.i.d. $\mathcal{N}(0,1)$ variables.

The optimal value of the problem is estimated by Defourny et al. (2013) at about 375; we refer to their paper for more details, in particular for the values of the parameters. The numerical experiments are performed with the STGM-EPs in Table 1 and for scenario trees with 125, 512, and 1000 scenarios (corresponding to the branching coefficients 5, 8, and 10, respectively).

Optimal selection of sample sizes:

The quality parameters are estimated using the procedure described in Section 3.3, for a computational time limit of two hours for each STGM-EP. Table 5 displays the optimal sample sizes K^* and M^*. The value $K^* \times M^*$, which is the number of out-of-sample scenarios used to test the corresponding STGM-EP, is larger for the couples involving the extension NN-AC, since it is computationally less costly than the other two extensions (it does not require to partition the whole support of the random parameters at every stage).

Quality parameters $p(t)$ and CR:

We compute the quality parameters $p(t)$ and CR for an extended decision policy of the form $\widetilde{\boldsymbol{x}} = (\widetilde{\boldsymbol{x}}_0, \widetilde{\boldsymbol{x}}_1, \widetilde{\boldsymbol{x}}_2, r_3)$, where r_3 is the optimal recourse function at stage 3 available analytically for this four-stage assembly problem. Although the use of a recourse policy was not initially introduced for the quality parameters $p(t)$ and CR, in this problem it is readily deducible from the constraints (68)-(69), hence we take it into account in the computation of $p(t)$ and CR. The optimal recourse

	125 scen.		512 scen.		1000 scen.	
	K^*	M^*	K^*	M^*	K^*	M^*
OQ–NN-AC	1	11×10^6	1	9×10^6	1	7×10^6
OQ–NN-AT	1	6×10^6	1	3×10^6	1	2×10^6
OQ–2NNW	1	4×10^6	1	2×10^6	1	2×10^6
RQMC–NN-AC	4613	376	873	491	298	474
RQMC–NN-AT	4783	155	825	301	286	315
RQMC–2NNW	4639	139	792	409	275	323
MC–NN-AC	5079	172	842	707	294	457
MC–NN-AT	4829	147	863	159	278	318
MC–2NNW	4764	118	786	430	280	327

Table 5: Optimal sample sizes K^* and M^* for a limit of 2h of computation for each STGM-EP. The values M^* are rounded-off to the nearest 10^6 for OQ.

decisions at stage 3 are given by

$$r_{3,k}(x_{\cdot 2}; \xi_{\cdot 3}) = \min\left(x_{2,k}, \max(0, b_{k,0} + \sum_{t=1}^{3} b_{k,t}\xi_t)\right), \quad \forall k \in \{1, \ldots, 5\}. \tag{71}$$

It follows that the probability of feasibility does not reduce from stage 2 to 3, i.e., we have $p(3) = p(2)$. Additionally, we note that all three extension procedures yield stage-1 decisions that satisfy the constraint (66), hence we also have $p(1) = 1$ for all STGM-EPs. We display in Table 6 the estimates of $p(2)$ and CR.

	125 scen.		512 scen.		1000 scen.	
	$p(2)$	CR	$p(2)$	CR	$p(2)$	CR
OQ–NN-AC	**1**	**366.1** (±0.4)	**1**	**370.7** (±0.5)	**1**	**372.8** (±0.5)
OQ–NN-AT	**0.986**	**366.3** (±0.6)	**0.986**	**370.9** (±0.8)	**0.985**	**371.9** (±0.9)
OQ–2NNW	0.402	352.3 (±1.2)	0.396	426.0 (±1.6)	0.394	365.1 (±1.8)
RQMC–NN-AC	**1**	**349.8** (±1.0)	**1**	**365.7** (±2.0)	**1**	**369.3** (±3.4)
RQMC–NN-AT	0.958	350.4 (±1.5)	0.958	367.3 (±2.6)	0.958	374.3 (±4.3)
RQMC–2NNW	0.334	3.4 (±2.6)	0.414	283.6 (±4.0)	0.414	351.3 (±7.6)
MC–NN-AC	**1**	**296.7** (±1.4)	**1**	**331.7** (±1.7)	**1**	**343.8** (±3.5)
MC–NN-AT	0.871	300.1 (±1.7)	0.873	340.2 (±3.8)	0.875	354.8 (±4.53)
MC–2NNW	0.356	93.0 (±3.0)	0.442	310.1 (±3.7)	0.446	383.2 (±7.1)

Table 6: Estimates of the quality parameters $p(2)$ and CR for an extended decision policy of the form $\tilde{\boldsymbol{x}} = (\tilde{\boldsymbol{x}}_0, \tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, r_3)$. Data in bold font single out the STGM-EPs that satisfy $p(2) \geq 0.98$. Parentheses in the CR-column include the 95% confidence intervals. In the $p(2)$-column this interval is not shown for the sake of clarity and because it is typically smaller than the last decimal displayed.

We note that the extension NN-AC satisfies *exactly* $p(2) = 1$, because the decisions that it yields always satisfy the constraint (67). The estimates in Table 6 clearly show the following hierarchy among the scenario-tree generation methods and extension procedures:

$$\text{OQ} > \text{RQMC} > \text{MC} \quad \text{and} \quad \text{NN-AC} > \text{NN-AT} > \text{2NNW}. \tag{72}$$

The hierarchy remains the same as the first case study for the scenario-tree generation methods. However, for the extension procedure, we have now that the nearest neighbor (NN) extensions yield better quality decisions than the extension with the 2 nearest neighbors (2NNW). The reason for this could be that the optimal decision functions at stage 1 and 2 are close to piecewise constant, and therefore the NN extension is more suitable for approximating them.

In particular, we observe that the couple OQ–NN-AC with 1000 scenarios yields a decision policy with 100% probability of feasibility and expected revenues estimated at 372.8 ± 0.5, hence only about 0.6% away from optimality (assuming it is equal to 375, as estimated by Defourny et al. (2011)). As a result, it is possible to build a decision policy at almost optimality for the four-stage multi-product assembly problem using the optimal quantization method for generating the scenario tree and the nearest-neighbor extension across children for extending the decisions out of the set of scenarios.

Expected value solution

We end this section by showing that the four-stage multi-product assembly problem should not be solved by the deterministic approach, known as the *expected value solution* (see Birge and Louveaux (1997), Chapter 4), that consists in solving the program (65)-(70) where $\boldsymbol{\xi}$ is substituted by its expectation (hence resulting in a deterministic program). Since the random parameters enter the constraints at stage 3, the decisions obtained by such approach are feasible from stage 0 to 2. At stage 3, we complete these decisions by the recourse functions (71) to obtain a feasible decision policy, which can then be compared with the decision policies built from the extension procedures. The expected revenues obtained by this approach (also known as the *expected result of using the expected value solution*) are estimated at 263 ± 1, with 95% confidence. We see that this value is much smaller than the expected revenues obtained by any of the scenario-tree generation method combined with the extension NN-AC (see Table 6).

5.4 Efficiency of the sample size selection technique

We end the numerical experiments by showing that the estimator $\widehat{\theta}_{K,M}$ defined in (29) and the optimal sample sizes selection introduced in Section 3 provide an efficient way for estimating the quality parameters. In particular, we want to show that it is more efficient that the classical estimator $\widetilde{\theta}_N$ that samples N times the random policy *and* the stochastic process, as opposed to $\widehat{\theta}_{K,M}$, which samples K times the random policy and $K \times M$ times the stochastic process. In the following, we use the notation introduced in Section 3.2 and 3.3.

Recall that we want to estimate an expectation of the form $\theta = \mathbb{E}[\phi(\boldsymbol{x}, \boldsymbol{\xi})]$. To this end, we define the estimator $\widetilde{\theta}_N$ as follows:

$$\widetilde{\theta}_N = \frac{1}{N} \sum_{n=1}^{N} \phi(x^n, \xi^n), \tag{73}$$

where $\{(x^n, \xi^n) \,|\, n = 1, \ldots, N\}$ is a set of N i.i.d. samples of $(\boldsymbol{x}, \boldsymbol{\xi})$. To compare the efficiency of $\widehat{\theta}_{K,M}$ and $\widetilde{\theta}_N$ for estimating θ, we compare their variances. They are given by

$$\mathrm{Var}(\widehat{\theta}_{K,M}) = \frac{\beta + \gamma(M-1)}{KM} \quad \text{and} \quad \mathrm{Var}(\widetilde{\theta}_N) = \frac{\beta}{N}, \tag{74}$$

where β and γ are defined in (31) and (32). More specifically, we want to compare $\mathrm{Var}(\widehat{\theta}_{K^*,M^*})$ and $\mathrm{Var}(\widetilde{\theta}_{N^*})$, where (K^*, M^*) is given by the optimal sample size selection technique described

in Section 3.3, and N^* is the number of sample points (x^n, ξ^n) that can be obtained within the same time limit τ. Since the computation of $\widetilde{\theta}_N$ takes $N(t_0 + t_1 + t_2)$ units of time, we simply have $N^* = \lfloor \tau/(t_0 + t_1 + t_2) \rfloor$.

The variances are compared in the context of the first case study, where the quality parameter $\mathbb{E}_x[Q(\overline{x})]$ was computed in Table 4 with a time limit of one hour for each scenario-tree generation method. Table 7 displays the ratio $\mathrm{Var}(\widetilde{\theta}_{N^*})/\mathrm{Var}(\widehat{\theta}_{K^*,M^*})$ for the RQMC and MC methods (the comparison is not relevant for the OQ method because $K^* = 1$). We see that the ratio ranges from about 8 to 16, hence $\mathrm{Var}(\widetilde{\theta}_{N^*})$ is typically one order of magnitude larger than $\mathrm{Var}(\widehat{\theta}_{K^*,M^*})$. This comparison shows that the estimator $\widehat{\theta}_{K^*,M^*}$ achieves a higher accuracy than $\widetilde{\theta}_{N^*}$ when the computational time is limited.

	5 scen.	20 scen.	40 scen.	80 scen.
RQMC	12.39	16.48	9.18	14.52
MC	7.84	10.11	14.06	12.25

Table 7: Estimates of $\mathrm{Var}(\widetilde{\theta}_{N^*})/\mathrm{Var}(\widehat{\theta}_{K^*,M^*})$.

6 Conclusion

In this paper, we introduced a quality evaluation framework, whose goal is to help the decision-maker to find the most suitable scenario-tree generation method and extension procedure (STGM-EP) to solve a given stochastic programming problem. From the scenario-tree optimal decisions, which are given for a finite set of scenarios only, the extension procedure builds a decision policy of the original problem. The framework consists of several quality parameters that assess the decisions yielded by a STGM-EP, and of several selection criteria that find the most suitable STGM-EP for a given problem in different settings. We focus in particular on the average-case setting, because we think that it is the most relevant when the problem is solved regularly with different data for the random parameters, which is the way scenario trees are often used in practice. The framework also includes the statistical tools (estimators, confidence intervals, sampling strategy) needed for an efficient application on a real-world problem. Overall, this newly introduced framework can be applied to a great deal of problems (two-stage or multistage, linear or nonlinear, continuous or integer variables) and to all scenario-tree generation methods, as far as we are aware.

We apply this framework on a two-stage newsvendor problem and a four-stage multi-product assembly problem. We demonstrate that simple extension procedures, such as the ones that set the decisions to the value of the nearest-neigbhor nodes in the scenario tree, provide good quality decisions at little computational cost. The application of the average-case selection criterion reveals that among the three tested methods, namely the Monte Carlo method, the randomized quasi-Monte Carlo method, and the optimal quantization method, the last one yields the highest quality decisions in both case studies. Randomized quasi-Monte Carlo method also yields good quality decisions, but it typically requires more scenarios to achieve a similar quality than the optimal quantization method. The quality of the Monte Carlo method is always statistically significantly below the other two methods for the small scenario sizes. The fact that the Monte Carlo method requires much more scenarios to provide acceptable quality decisions confirms that it is not suitable for multistage stochastic programming.

The quality evaluation framework was developed for stochastic programming problems with an objective function given by an expectation. We think that future work should look for a generalization to objective functions that also include some risk measures. More extension procedures should

also be investigated. Several techniques other than interpolation and extrapolation can be considered, such as regression and curve fitting. Another important future work would be the quality evaluation of decision policies on the so-called *rare events* (also known as *black swan events*). Methods that build scenario trees by taking into account the stochastic process, but not the variation of the revenue function, may not incorporate those events in the tree. This result in overoptimistic policies that may provide decisions with disastrous consequences should one of these events occur.

Overall, we hope that the approach introduced in this paper will bring more attention to the importance of the evaluation of decision policies and scenario-tree generation methods. We believe that quality evaluation techniques should eventually be included in every stochastic optimization software to help practitioners making better decisions.

A Appendix: Notation

Notation	Description
T	optimization horizon
$\boldsymbol{\xi}_t$	random vector at stage t
$\boldsymbol{\xi}$ or $(\boldsymbol{\xi}_1, ..., \boldsymbol{\xi}_T)$	stochastic process
$\boldsymbol{\xi}_{..t}$	components from stage 0 to stage t of $\boldsymbol{\xi}$
ξ_t (resp. ξ; resp. $\xi_{..t}$)	a realization of $\boldsymbol{\xi}_t$ (resp. $\boldsymbol{\xi}$; resp. $\boldsymbol{\xi}_{..t}$)
Ξ_t (resp. Ξ; resp. $\Xi_{..t}$)	support of $\boldsymbol{\xi}_t$ (resp. $\boldsymbol{\xi}$; resp. $\boldsymbol{\xi}_{..t}$)
d	number of random parameters revealed at each period
s	number of decisions made at each stage
$q(\cdot;\cdot)$	revenue function
$Q(\cdot)$	expected revenue function
$Q(x^*)$	optimal value of the stochastic program
x_t	decision function if $t \geq 1$; decision vector if $t = 0$
x or $(x_0, ..., x_T)$	decision policy
$x_{..t}$	components from stage 0 to stage t of x
x^*	optimal policy of the stochastic program
X_t	feasible decision set at stage t
$\widetilde{x}$	extended decision policy
$\overline{x}$	feasible extended decision policy
$\mathcal{N}$	node set of the scenario tree
$\mathcal{E}$	edge set of the scenario tree
$\mathcal{N}^+$	set of nodes minus the root
$\mathcal{N}_t$	node set at stage t
n_0	root node
$a(n)$	ancestor node of n
$C(n)$	set of children nodes of n
ζ^n	discretization vector at node n
$\zeta^{..n}$	discretization sequence leading to node n
w^n	weight of node n with respect to its siblings
W^n	weight of node n with respect to the whole scenario tree
$\widehat{Q}^*$	optimal value of a deterministic program
$\widehat{x}^n$	decision vector at node n
$\widehat{x}^{..n}$	sequence of decision vectors leading to node n
$\widehat{x}^{*n}$	optimal decision vector at node n

References

Bayraksan, G. and Morton, D. P. (2009). Assessing solution quality in stochastic programs via sampling. In *Decision Technologies and Applications*, pages 102–122. Informs.

Ben-Tal, A., El Ghaoui, L., and Nemirovski, A. (2009). *Robust optimization*. Princeton University Press.

Birge, J. R. and Louveaux, F. (1997). *Introduction to stochastic programming*. Springer, New York.

Chen, M. and Mehrotra, S. (2008). Epi-convergent scenario generation method for stochastic problems via sparse grid. *Stochastic Programming E-Print Series*, (7).

Chen, M., Mehrotra, S., and Papp, D. (2015). Scenario generation for stochastic optimization problems via the sparse grid method. *Computational Optimization and Applications*, 62(3):669–692.

Chiralaksanakul, A. and Morton, D. P. (2004). Assessing policy quality in multi-stage stochastic programming. *Stochastic Programming E-Print Series*, (12).

Defourny, B., Ernst, D., and Wehenkel, L. (2011). Multistage stochastic programming: A scenario tree based approach. *Decision Theory Models for Applications in Artificial Intelligence: Concepts and Solutions: Concepts and Solutions*, page 97.

Defourny, B., Ernst, D., and Wehenkel, L. (2013). Scenario trees and policy selection for multistage stochastic programming using machine learning. *INFORMS Journal on Computing*, 25(3):488–501.

Drew, S. S. and Homem-de Mello, T. (2006). Quasi-Monte Carlo strategies for stochastic optimization. In *Proceedings of the 38th conference on Winter simulation*, pages 774–782. Winter Simulation Conference.

Dupačová, J., Gröwe-Kuska, N., and Römisch, W. (2003). Scenario reduction in stochastic programming. *Mathematical programming*, 95(3):493–511.

Dyer, M. and Stougie, L. (2006). Computational complexity of stochastic programming problems. *Mathematical Programming*, 106(3):423–432.

Edirisinghe, N. (1999). Bound-based approximations in multistage stochastic programming: Nonanticipativity aggregation. *Annals of Operations Research*, 85:103–127.

Frauendorfer, K. (1996). Barycentric scenario trees in convex multistage stochastic programming. *Mathematical Programming*, 75(2):277–293.

Hanasusanto, G. A., Kuhn, D., and Wiesemann, W. (2016). A comment on "computational complexity of stochastic programming problems". *Mathematical Programming*, 159(1):557–569.

Heitsch, H. and Römisch, W. (2009). Scenario tree modeling for multistage stochastic programs. *Mathematical Programming*, 118(2):371–406.

Hilli, P. and Pennanen, T. (2008). Numerical study of discretizations of multistage stochastic programs. *Kybernetika*, 44(2):185–204.

Høyland, K., Kaut, M., and Wallace, S. W. (2003). A heuristic for moment-matching scenario generation. *Computational Optimization and Applications*, 24(2-3):169–185.

Høyland, K. and Wallace, S. W. (2001). Generating scenario trees for multistage decision problems. *Management Science*, 47(2):295–307.

Kaut, M. and Wallace, S. W. (2007). Evaluation of scenario-generation methods for stochastic programming. *Pacific Journal of Optimization*, 3(2):257–271.

Koivu, M. (2005). Variance reduction in sample approximations of stochastic programs. *Mathematical programming*, 103(3):463–485.

Kouwenberg, R. (2001). Scenario generation and stochastic programming models for asset liability management. *European Journal of Operational Research*, 134(2):279–292.

Küchler, C. and Vigerske, S. (2010). Numerical evaluation of approximation methods in stochastic programming. *Optimization*, 59(3):401–415.

L'Ecuyer, P. and Lemieux, C. (2000). Variance reduction via lattice rules. *Management Science*, 46(9):1214–1235.

Leövey, H. and Römisch, W. (2015). Quasi-Monte Carlo methods for linear two-stage stochastic programming problems. *Mathematical Programming*, 151(1):315–345.

Louveaux, F. (1998). An introduction to stochastic transportation models. In *Operations Research and Decision Aid Methodologies in Traffic and Transportation Management*, pages 244–263. Springer.

Mak, W.-K., Morton, D. P., and Wood, R. K. (1999). Monte carlo bounding techniques for determining solution quality in stochastic programs. *Operations Research Letters*, 24(1):47–56.

Pennanen, T. (2005). Epi-convergent discretizations of multistage stochastic programs. *Mathematics of Operations Research*, 30(1):245–256.

Pennanen, T. and Koivu, M. (2002). Integration quadratures in discretization of stochastic programs. *Stochastic Programming E-Print Series*, (11).

Pflug, G. C. (2001). Scenario tree generation for multiperiod financial optimization by optimal discretization. *Mathematical programming*, 89(2):251–271.

Pflug, G. C. and Pichler, A. (2012). A distance for multistage stochastic optimization models. *SIAM Journal on Optimization*, 22(1):1–23.

Pflug, G. C. and Pichler, A. (2015). Dynamic generation of scenario trees. *Computational Optimization and Applications*, 62(3):641–668.

Powell, W. B. and Topaloglu, H. (2003). Stochastic programming in transportation and logistics. In Ruszczyński, A. and Shapiro, A., editors, *Handbooks in operations research and management science: Stochastic Programming*, volume 10, pages 555–635. Elsevier.

Proulx, S. (2014). Génération de scénarios par quantification optimale en dimension élevée. Master's thesis, École Polytechnique de Montréal.

Rockafellar, R. T. and Wets, R. J. (1974). Continuous versus measurable recourse in n-stage stochastic programming. *Journal of Mathematical Analysis and Applications*, 48(3):836–859.

Ruszczyński, A. and Shapiro, A. (2003). Stochastic programming models. In Ruszczyński, A. and Shapiro, A., editors, *Handbooks in operations research and management science: Stochastic Programming*, volume 10, pages 1–64. Elsevier.

Schultz, R. (2003). Stochastic programming with integer variables. *Mathematical Programming*, 97(1):285–309.

Schultz, R., Nowak, M. P., Nürnberg, R., Römisch, W., and Westphalen, M. (2003). Stochastic programming for power production and trading under uncertainty. In *Mathematics—Key Technology for the Future*, pages 623–636. Springer.

Shapiro, A. (2003). Monte Carlo sampling methods. In Ruszczyński, A. and Shapiro, A., editors, *Handbooks in operations research and management science: Stochastic Programming*, volume 10, pages 353–425. Elsevier.

Shapiro, A. and Homem-de Mello, T. (1998). A simulation-based approach to two-stage stochastic programming with recourse. *Mathematical Programming*, 81(3):301–325.

Sloan, I. H., Kuo, F. Y., and Joe, S. (2002). Constructing randomly shifted lattice rules in weighted sobolev spaces. *SIAM Journal on Numerical Analysis*, 40(5):1650–1665.

Wallace, S. W. and Fleten, S.-E. (2003). Stochastic programming models in energy. In Ruszczyński, A. and Shapiro, A., editors, *Handbooks in operations research and management science: Stochastic Programming*, volume 10, pages 637–677. Elsevier.

Yu, L.-Y., Ji, X.-D., and Wang, S.-Y. (2003). Stochastic programming models in financial optimization: A survey. *Advanced Modeling and Optimization*, 5(1).